Angewandte Reaktionskinetik mit MATLAB – Entwicklung einer Programmsammlung

Von der Gemeinsamen Naturwissenschaftlichen Fakultät
der Technischen Universität Carolo-Wilhelmina
zu Braunschweig

zur Erlangung des Grades eines
Doktors der Naturwissenschaften
(Dr. rer. nat.)

genehmigte
D i s s e r t a t i o n

von
Arne Ohrenberg
aus Bad Oldesloe

1. Referent: Prof. Dr. A. Löwe
2. Referent: Prof. Dr. A. Schumpe
eingereicht am: 11. November 1999
mündliche Prüfung (Disputation) am: 28. Januar 2000
Druckjahr: 2000

Meinen Eltern

Meinen Lehrern

Danksagung

Die vorliegende Arbeit wurde in der Zeit von Oktober 1996 bis Oktober 1999 am Institut für Technische Chemie, Abteilung Reaktionstechnik, der Technischen Universität Carolo-Wilhelmina zu Braunschweig angefertigt.

Allen, die zum Gelingen dieser Arbeit beigetragen haben, möchte ich herzlich danken, insbesondere

meinem Doktorvater, Herrn Univ.-Prof. Dr. A. Löwe, für die Anregung zu dieser Arbeit und seine Betreuung,

Herrn Univ.-Prof. Dr. A. Schumpe für die Übernahme des Korreferats

und allen Mitarbeitern der Arbeitsgruppe für ihre ständige Diskussions- und Hilfsbereitschaft, besonders Herrn Dr. Karsten Horn und Herrn Dr. Dirk-Chr. Eiting.

Ferner gilt mein Dank Herrn Univ.-Prof. Dr. G. Emig und der DECHEMA e.V. für die freundliche Überlassung von Unterlagen zum Kurs „Planung und Auswertung von Versuchen zur Erstellung mathematischer Modelle“ (Bearbeitungsstand 1974).

Zusammenfassung

Im Rahmen der vorliegenden Arbeit ist eine Programmsammlung zur Planung und Auswertung von reaktionskinetischen Versuchen mit und für das Softwarepaket MATLAB entwickelt worden. Dabei liegt das Ziel dieser Arbeit in der Verknüpfung der theoretischen Grundlagen zur reaktionskinetischen Modellierung mit den mathematischen Möglichkeiten von MATLAB. Auf diese Weise soll dem steigenden Bedarf an rechnergestützten Modellierungstechniken in der Reaktionstechnik entsprochen werden.

Der Einsatz von MATLAB hat sich dabei als vorteilhaft erwiesen, da es für mathematische Hochleistungsrechnungen ausgelegt ist und über eine relativ einfache, aber effizient angelegte Programmierumgebung verfügt.
MATLAB selbst ist bisher nicht für die gezielte Bearbeitung reaktionskinetischer Fragestellungen ausgestattet.

Inhaltlich stellt diese Programmsammlung Verfahren zur Parameterschätzung, zur statistischen Beurteilung von Parameterschätzungen und zur Modellunterscheidung sowie zur Parameterpräzisierung zur Verfügung. Dabei liegt ein Vorzug der vorliegenden Programmsammlung in der Möglichkeit, Optimierverfahren und Schätzkriterien relativ einfach zu ergänzen.
Ferner sind für die Anwendung der Programmsammlung spezielle, numerische Aspekte berücksichtigt worden, wie z. B. die Ermittlung von Startschätzwerten.
Bei der Erstellung der vorliegenden Programme sind die Möglichkeiten von MATLAB gezielt genutzt, gegebenenfalls den Bedürfnissen angepaßt oder durch zusätzliche numerische Verfahren ergänzt worden.

Die Einsatzfähigkeit der entwickelten Programme ist anhand gut dokumentierter Fremdbeispiele untersucht worden, die häufig auftretende reaktionskinetische Fragestellungen beinhalten. Die Leistungsfähigkeit der Programme ergibt sich dabei weitgehend aus der Leistungsfähigkeit der verwendeten MATLAB-Verfahren.

Die entwickelte Programmsammlung stellt eine fast eigenständige Ergänzung zur MATLAB-Grundversion dar. Bezüglich ihrer Handhabung und Erscheinung fügt sie sich in das Konzept von MATLAB ein. Aus Gründen der Vollständigkeit und der Einheitlichkeit treten jedoch teilweise inhaltliche Überschneidungen mit verschiedenen MATLAB-Toolboxen auf.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung und Aufgabenstellung

In den letzten Jahren haben rechnergestützte Modellierungs- und Simulationstechniken an erheblicher Bedeutung für die Chemische Technik gewonnen [1]. Die wichtigsten Einsatzgebiete dieser Techniken sind die Auslegung von Produktionsanlagen, die Optimierung von Prozeßführungen und die Übertragung des Labormaßstabs auf den technischen Maßstab. Der Vorteil bei der Anwendung derartiger Simulationsverfahren liegt gegenüber den herkömmlichen experimentellen Arbeiten in einer möglichen Verringerung des Arbeitsaufwands und einer Kostensenkung. Die Voraussetzung für die Anwendbarkeit von rechnergestützten Modellierungs- und Simulationstechniken ist in der Entwicklung leistungsfähiger Rechnertechnologien und in der Verbesserung und Neuentwicklung numerischer Algorithmen zu sehen. Auf der Grundlage dieser Entwicklungen sind verschiedene Programmsysteme für numerische Hochleistungsrechnungen entstanden.
Ein System, das sich dabei als besonders einsatzfähig erwiesen hat, ist MATLAB™ von der Firma MATHWORKS [2,3,4,5].
MATLAB besteht in seiner Grundversion aus den wichtigsten numerischen Verfahren und mathematischen Funktionen und kann durch Ergänzungssysteme, sogenannten *toolboxes*, den individuellen Bedürfnissen angepaßt werden. Allerdings verfügt MATLAB bisher noch nicht über eine speziell auf die Chemische Technik bezogene Toolbox [6].

Die Motivation für die vorliegende Arbeit liegt in der Verknüpfung der Vorzüge von MATLAB mit dem Bedarf an qualitativ hochwertigen Programmen zur einfachen und schnellen Bearbeitung von technisch-chemischen Fragestellungen. Ein wichtiges Teilgebiet dieser Fragestellungen ist die Planung und Auswertung von Versuchen zur Reaktionskinetik, denn detaillierte Kenntnisse bezüglich kinetischer Modelle stellen eine wesentliche Grundlage für die Simulation und Optimierung chemischer Prozesse dar.
Die kinetische Modellierung ist dabei durch Parameterschätzungen in hochgradig nichtlinearen Systemen und durch die Schwierigkeit einer eindeutigen Modellauswahl geprägt [7]. Dieses erfordert teilweise die Anwendung spezieller Lösungsstrategien und Fachkenntnisse, wodurch eine eigenständige Behandlung dieser Thematik durchaus gerechtfertigt ist.
Aus diesen Gründen ist im Rahmen dieser Arbeit eine anwendungsorientierte MATLAB-Programmsammlung unter folgender Aufgabenstellung entwickelt worden:

- Erstellung von Programmen zur Planung und Auswertung von reaktionskinetischen Versuchen,
- Entwicklung als eigenständige Ergänzung zur MATLAB-Grundversion und
- Beurteilung der Einsatz- und Leistungsfähigkeit von MATLAB in der Reaktionskinetik.

Thematisch umfaßt diese Programmsammlung den in Abb.1-1 dargestellten allgemeinen Ablauf einer mathematischen Modellierung, der in dieser Weise auch für reaktionskinetische Modelle gültig ist.

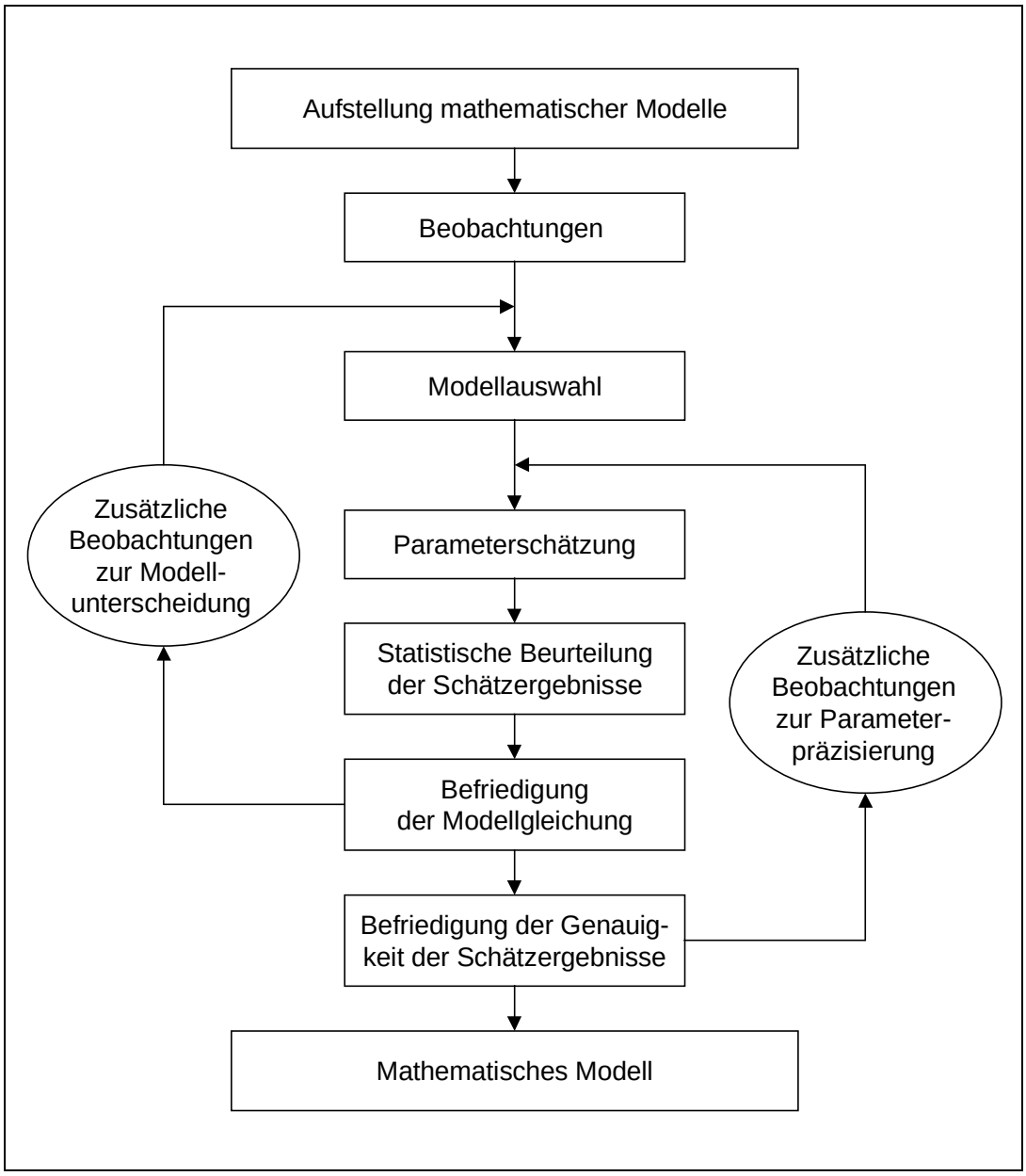

Abb. 1-1. Ablauf einer mathematischen Modellierung nach [8]

Die Programmsammlung beinhaltet Funktionen und Programme zur Parameterschätzung, Modelldiskriminierung und Versuchsplanung, wobei zusätzliche Schwerpunkte auf den Bereich der statistischen Beurteilung von Parameterschätzungen und auf die Auswahl von Startschätzwerten bei der numerischen Optimierung gelegt worden sind. Fragestellungen zum Modellaufbau werden nicht behandelt.

Die Programme und Funktionen sind in Kombination mit den von MATLAB zur Verfügung gestellten Möglichkeiten entwickelt worden. Dabei sind die Funktionen nach dem *matlab*-typischen Baukastenprinzip angelegt worden, so daß sie problemlos in MATLAB-Programmierungen eingebunden werden können und sich somit in das Konzept von MATLAB einfügen.

Die hier aufgeführten Programme erheben in ihrer Auswahl nicht den Anspruch auf Vollständigkeit bezüglich aller relevanten Verfahren und Methoden, sondern stellen Möglichkeiten bereit, reaktionskinetische Problemstellungen in den genannten Bereichen zu bearbeiten.

2 Grundlagen der Parameterschätzung

Ein thematischer Schwerpunkt der hier vorliegenden Arbeit ist die Parameterschätzung mittels Regression, weshalb zunächst die theoretischen Grundlagen näher erläutert werden sollen.

2.1 Einführung

Ein Prozeß, wie z. B. ein naturwissenschaftliches Experiment, ist im allgemeinen durch unterschiedliche Beobachtungsgrößen oder Variablen gekennzeichnet. Man unterscheidet zwischen unabhängigen und abhängigen Variablen. Zu den unabhängigen Variablen gehören die Einstellgrößen als extern veränderbare Größen und die Störgrößen, die nicht von außen beeinflußt werden können. Als abhängige Variablen werden die Antworten des Prozesses betrachtet; sie stellen das Ergebnis von Änderungen der extern vorgegebenen Größen dar.
Die abhängigen und unabhängigen Variablen sind in der mathematischen Beschreibung ihres Zusammenhangs über einen Satz von Parametern gemäß

$$\boldsymbol{\eta} = \mathbf{f}(\boldsymbol{\beta}, \mathbf{x}) \tag{2-1}$$

miteinander verknüpft. Dabei ist $\boldsymbol{\eta}$ der Vektor der abhängigen Variablen, $\boldsymbol{\beta}$ der Vektor der Parameter und $\mathbf{x}$ der Vektor der unabhängigen Variablen.
Eine Beobachtungsgröße ist im allgemeinen mit einem Meßfehler $\boldsymbol{\varepsilon}$ behaftet, der den Unterschied zwischen dem wahren Wert $\boldsymbol{\eta}$ und dem Beobachtungswert $\mathbf{y}$ gemäß Gl. (2-2) darstellt.

$$\mathbf{y} = \boldsymbol{\eta} + \boldsymbol{\varepsilon} \tag{2-2}$$

Ziel einer Regressionsanalyse ist es, auf der Basis der Beobachtungswerte einen Parametersatz zu finden, für den die Differenzen zwischen Beobachtungswerten und Modellantworten möglichst klein sind. Diese Differenzen werden als Residuen bezeichnet. Setzt man die Schätzwerte der Parameter in Gl.(2-1) ein, so erhält man die Regressionsgleichung

$$\hat{\mathbf{y}} = \mathbf{f}(\mathbf{b}, \mathbf{x}) \tag{2-3}$$

Hierbei stellt der Vektor $\hat{\mathbf{y}}$ einen Schätzwert für den Vektor der wahren Werte $\boldsymbol{\eta}$ auf der Grundlage der geschätzten Parameter $\mathbf{b}$ dar. Weil die wahren Werte aber nie bekannt sind, beschreibt Gl. (2-3) lediglich mit relativ hoher Wahrscheinlichkeit die tatsächlichen Zusammenhänge [8,9].

2.2 Schätzmethoden

Die Menge der Beobachtungswerte wird in der Regressionsanalyse als Stichprobe einer Gesamtheit aufgefaßt. Aus dieser Menge werden Näherungswerte für die unbekannten Parameter ermittelt. Die Beobachtungswerte lassen sich in ihrem Auftreten durch Verteilungsfunktionen beschreiben, deren Verlauf auch durch die Parameter bestimmt wird.
In der Behandlung der chemischen Kinetik entspricht der abstrakt eingeführte Begriff des Prozesses einem naturwissenschaftlichen Experiment. Damit sind die Beobachtungsgrößen Meßgrößen, die Ungenauigkeiten aufweisen und im allgemeinen der Normalverteilung gehorchen. Die Normalverteilung hat zwei wichtige Kenngrößen: den Mittelwert und die Varianz. Der Mittelwert gibt den wahrscheinlichsten Wert aus einer Anzahl an Wiederholungsmessungen an. Mit der Varianz kann der Bereich klassifiziert werden, in dem das Auftreten von Meßdaten besonders wahrscheinlich ist.
Bei Regressionsproblemen mit angenommener Normalverteilung ist der Mittelwert eine bekannte Funktion der Modellparameter. Auf dieser Grundlage können die Parameter und die Varianz der Verteilung geschätzt werden. Die so ermittelten Parameter stellen den wahrscheinlichsten Parametersatz dar.
Die Schätzmethoden können sich in der Berücksichtigung verschiedener Annahmen und damit auch in den Ergebnissen der Schätzung unterscheiden. In der Programmsammlung stehen verschiedene Schätzfunktionen zur Verfügung, wobei die Wahl der Bewertungsfunktion und das Vertrauen in die Ergebnisse von der konkreten Problemstellung abhängen.
Bei der Auswahl der Schätzfunktionen sollte berücksichtigt werden, daß die Kriterien möglichst erwartungstreu, wirksam und konsistent sind. Die Schätzfunktion eines Parameters ist erwartungstreu, wenn der Erwartungswert gleich dem unbekannten Parameter ist; sie ist wirksam, wenn der Schätzwert mit großer Wahrscheinlichkeit in der Nähe des tatsächlichen Werts liegt, und konsistent, wenn mit wachsendem Stichprobenumfang der Erwartungswert der Varianz gegen Null strebt.
Allgemein unterscheidet man drei Arten von Schätzmethoden: die Methode der kleinsten Fehlerquadrate, die Maximum-Likelihood-Methode und die Bayessche Methode [10,11]. Diese Methoden werden nachfolgend kurz vorgestellt.

Die Methode der kleinsten Fehlerquadrate kann man als Minimierung der Summe der Residuenquadrate auffassen, wobei die Parameter über die

Berechnung der Modellantworten nach Gl.(2-3) eingehen. Für ein System von r Antworten an n Versuchspunkten gilt dann

$$\sum_{j=1}^{r}\sum_{i=1}^{n}\left(Y_{ji}-\hat{Y}_{ji}\right)^2 \rightarrow Min \tag{2-4}$$

Hierbei ist Y_{ij} die (i,j)-te Komponente der Matrix der Antworten $\mathbf{Y}$. Die Doppelsumme berücksichtigt alle auftretenden Residuen. Dieses Kriterium kann variiert werden, indem für die einzelnen Versuchspunkte Gewichte eingeführt werden [11].
Insgesamt ist die Methode der kleinsten Fehlerquadrate jedoch ein Spezialfall der Maximum-Likelihood-Methode.

Bei der Maximum-Likelihood-Methode geht man davon aus, daß die Wahrscheinlichkeitsdichte der Beobachtungswerte nur in ihrer Form bzw. Struktur gegeben ist. Die Aufgabe besteht nun darin, die unbekannten Parameter und die Varianz zu schätzen. Dieses geschieht, indem die Likelihood-Funktion Gl.(2-5), die das Produkt über die Wahrscheinlichkeiten aller auftretenden Antworten darstellt, maximiert wird.

$$L(\boldsymbol{\beta} \mid \mathbf{Y}) = \prod_{j=1}^{r}\prod_{i=1}^{n} p(Y_{ji}) \tag{2-5}$$

Die Likelihood-Funktion für normalverteilte Fehler mit dem Mittelwert Null, den Varianz-Kovarianz-Matrizen der Meßfehler in den Antworten $\mathbf{V}_i$ und den Antwortvektoren $\mathbf{y}_i$ bzw. $\hat{\mathbf{y}}_i$ an den verschiedenen Versuchspunkten i entspricht nach [7].

$$L(\boldsymbol{\beta}, \mathbf{V}_1, \ldots, \mathbf{V}_n) = (2\pi)^{-\frac{r\cdot n}{2}} \cdot \prod_{i=1}^{n}\left(\det(\mathbf{V}_i)\right)^{-\frac{1}{2}} \cdot \exp\left(-\frac{1}{2}\sum_{i=1}^{n}\left(\mathbf{y}_i-\hat{\mathbf{y}}_i\right)^T \mathbf{V}_i^{-1}\left(\mathbf{y}_i-\hat{\mathbf{y}}_i\right)\right) \tag{2-6}$$

Eine Maximierung der Likelihood-Funktion bedeutet hier eine Minimierung des Arguments der Exponentialfunktion, wodurch sich das Schätzkriterium auf eine allgemeine Form der gewichteten Quadratsumme reduzieren läßt [9].

Bei der Bayes-Methode werden zwar die zu bestimmenden Parameter wie in der Maximum-Likelihood-Methode als Zufallsvariablen betrachtet. Allerdings wird hier zur Parameterschätzung die Wahrscheinlichkeit $p(\mathbf{Y},\boldsymbol{\beta})$ für das gemeinsame Eintreten aller Ereignisse, also der Antworten und der Parameter, maximiert.

Diese Wahrscheinlichkeit ist jedoch unbekannt, weshalb man unter Berücksichtigung von Randverteilungen und bedingten Wahrscheinlichkeiten $p(\mathbf{Y}|\boldsymbol{\beta})$ folgende Beziehung zur Schätzung verwendet [8]:

$$p(\mathbf{Y}|\mathbf{b}) \cdot p_0(\mathbf{b}) \geq p(\mathbf{Y}|\boldsymbol{\beta}) \cdot p_0(\boldsymbol{\beta}) \tag{2-7}$$

Das eigentliche Schätzkriterium besteht nun darin, die rechte Seite von Gl.(2-7) möglichst groß zu wählen. Hierbei ist $p_0(\boldsymbol{\beta})$ der Schätzwert der a-priori-Wahrscheinlichkeit der Parameter und muß im allgemeinen durch Vorversuche bestimmt werden. Ist sie unbekannt, entspricht dieses Kriterium einer Maximum-Likelihood-Schätzung.

Wegen der Vielzahl der möglichen Zielfunktionen wird im Rahmen dieser Arbeit kein Anspruch auf Vollständigkeit erhoben, sondern es werden lediglich die gebräuchlichsten Kriterien in den Programmen eingesetzt. Der Aufbau der Programme ist jedoch so angelegt, daß Schätzfunktionen nach Bedarf relativ einfach ergänzt werden können (vgl. Kap. 3.5.2).

2.3 Numerische Aspekte der Parameterschätzung

Zur Schätzung von Parametern wird im allgemeinen eine Optimierung durchgeführt, die das Ziel hat, das globale Optimum einer Schätzfunktion zu finden. Bei umfangreichen Systemen hängen damit die Ergebnisse einer Schätzung nicht nur von der Wahl des Bewertungskriteriums ab, sondern auch von den eingesetzten numerischen Verfahren, die eine mehrdimensionale Optimierung großer Datenmengen erst ermöglichen. Mit der Verwendung numerischer Methoden wird die Parameterschätzung um zwei Problemstellungen erweitert: Zum einen müssen die Verfahren auf das Problem anwendbar sein, zum anderen müssen geeignete Einstellungen und insbesondere geeignete Startwerte für das Optimierverfahren gewählt werden [4].
MATLAB stellt hierfür vor allem mit der Erweiterung durch die *Optimization Toolbox* hochwertige Verfahren zur Verfügung, aber auch das in der Grundausstattung über die Funktion *fmins* eingebundene Simplex-Verfahren ist relativ leistungsstark, wie sich in den Untersuchungen im Rahmen dieser Arbeit gezeigt hat.

2.4 Aspekte der statistischen Beurteilung

Um das Vertrauen in geschätzte Parameter zu beurteilen, ist die Varianz-Kovarianz-Matrix der Fehler von zentraler Bedeutung. Da die wahren Fehler

nicht bekannt sind, wird diese Matrix im allgemeinen unter Verwendung der Residuen geschätzt. Für ein Mehrfachantwortsystem geben ihre Diagonalelemente die Schätzwerte der Fehlervarianzen der einzelnen Antworten an. Ist die Varianz-Kovarianz-Matrix für jeden Versuchspunkt bekannt, repräsentieren die Diagonalelemente die Varianz der einzelnen Antworten an den Versuchspunkten. Die Nichtdiagonalelemente spiegeln die Korrelation der Fehler untereinander wider.

Mit Hilfe dieser Varianz-Kovarianz-Matrix ist die Abschätzung von Vertrauensbereichen der Parameter möglich. Dabei wird zwischen individuellen und gemeinsamen Vertrauensbereichen unterschieden [8,9].
Die individuellen Vertrauensbereiche geben Intervalle an, in denen der jeweilige Parameter mit einer bestimmten Wahrscheinlichkeit liegt. Zu ihnen gehört die Standardabweichung mit einem Vertrauen von 68 %. Die Standardabweichungen werden aus den Diagonalelementen der Varianz-Kovarianz-Matrix der Schätzung erhalten und können zur Ermittlung für die *t*-verteilten individuellen Vertrauensbereiche herangezogen werden. Diese Konfidenzintervalle sind die kleinsten zulässigen Vertrauensintervalle. Im Gegensatz dazu stellen die Tangentialebenenvertrauensintervalle die maximalen individuellen Vertrauensintervalle dar, die als *F*-verteilte Größen über die Tangentialebenen an den gemeinsamen ellipsoiden Vertrauensbereichen ermittelt werden.
Bei der Ermittlung der gemeinsamen Vertrauensbereiche nimmt man an, daß alle Parameter eines Modells gleichzeitig den Zusammenhang zwischen den beobachteten Werten verursacht haben. Dabei liegt die Vorstellung zugrunde, daß die Schätzung der Parameter nicht unabhängig voneinander erfolgt und somit eine Korrelation der Parameter vorliegt. Im Parameterraum stellen sich die gemeinsamen Vertrauensbereiche je nach Berechnungsgrundlage als Hyperellipsoide oder Hyperkörper dar.

3 Einführung in die Programmsammlung

3.1 Einführung in MATLAB

MATLAB ist ein u. a. von WINDOWS unterstütztes Softwarepaket für numerische Hochleistungsrechnungen und Simulationstechniken sowie deren graphische Umsetzung. Es hat sich seit 1977 aus den damaligen FORTRAN-Bibliotheken LINPACK und EISPACK entwickelt und bietet heute umfassende Lösungen in den Bereichen Regelungssysteme und Simulation, digitale Signalverarbeitung, Bildverarbeitung und Finanzwesen an [12,13].
Der Begriff MATLAB steht dabei für *MATrix LABoratory* und bezeichnet damit eine wesentliche Stärke dieses Systems, die in der Matrizen- und Feldverarbeitung liegt. Dadurch wird die Entwicklungszeit bei der Verarbeitung großer Datenmengen auf einen Bruchteil vergleichbarer C- oder FORTRAN-Programme reduziert.
Die Befehlseingabe wird in MATLAB über ein typisches WINDOWS-Fenster, das *command window*, geregelt. Die in oder für MATLAB entwickelten Programme werden als *M-File* bezeichnet und können nicht nur vom Hauptfenster, sondern auch von jedem anderen *M-File* aus aufgerufen und verwendet werden, sofern sie in für den Zugriff durch MATLAB freigegebene Verzeichnisse abgelegt worden sind. Zur Erstellung derartiger M-Dateien steht ab der Version 5.0 ein gut handhabbarer Editor zur Verfügung. Sind diese *M-Files* als sogenannte *functions* deklariert und aufgebaut, können sie wie herkömmliche MATLAB-Funktionen eingesetzt und eingebunden werden. Die *functions* stellen Unterprogramme dar, die bei externer Vorgabe bestimmter Argumente die dazugehörenden Ergebnisse liefern. Die dafür notwendigen Rechnungen sind in ihren Auswirkungen lokal auf das Unterprogramm beschränkt. Durch diese Form der Programm- und Funktionsverwaltung ist die Ergänzung durch und die Einbindung von selbstentwickelten Funktionen relativ einfach.
Bei der Programmierung in MATLAB wird eine einfache *basic*-ähnliche Sprache benutzt, die über Programmierelemente wie diverse Schleifen- und Schalterbefehle verfügt. Besonders hervorzuheben ist hierbei, daß ab den Versionen 5.x die Möglichkeit besteht, Variablen dem Typ *cell array* zuzuordnen. Diese *cell arrays* sind Felder, deren Komponenten jeweils einem beliebigen Variablentyp angehören können. Über derartige *cell arrays* ist eine unkomplizierte Datenübertragung und –verarbeitung möglich.
Grundsätzlich können Programme und Funktionen in MATLAB auch über ein *graphical user interface* eingebaut werden, wodurch die Programmaktivierung und –verknüpfung über *windows*-typische Schaltoberflächen erfolgen kann [14].

Neben der einfachen Programmierung und einer großen Anzahl vorhandener mathematischer Funktionen sei noch auf die guten graphischen Darstellungs- und Bearbeitungsmöglichkeiten, die von der 2D- und 3D-Darstellung bis zum implementierten Höhenplotprogramm reichen, und auf die umfangreiche Sammlung numerischer Verfahren für Iterationen und Integrationen in MATLAB hingewiesen [15]. Außerdem stellt MATLAB die Möglichkeit bereit, externe C- oder FORTRAN-Programme einzubinden [16].
Die Ergänzung der Grundversion von MATLAB ist durch eine Vielzahl von *toolboxes* möglich, durch die das System speziellen Bedürfnissen angepaßt werden kann. Auf einige Toolboxen, die für die vorliegende Programmsammlung von Bedeutung sind, soll kurz eingegangen werden

- *Statistics Toolbox*
  Sammlung statistischer Funktionen und ihrer Anwendungen: Berechnung von Verteilungsfunktionen, statistische Verfahren und Techniken zur Versuchsauswertung und –planung [17]
- *Optimization Toolbox*
  Bereitstellung unterschiedlicher Optimierverfahren (vgl. Tab.3-2) [18]
- *Symbolic Toolbox (Extended Symbolic Toolbox)*
  Methoden zur symbolischen Verarbeitung mathematischer Gleichungen und Zugriff auf MAPLE [19]

3.2 Voraussetzungen

Die hier vorgestellten Programme sind für MATLAB 5.0 entwickelt und auf 5.2.1 angepaßt worden. Eine Übertragung der Programme auf die aktuelle Version 5.3 ist in Anhang B beschrieben.

Für die Anwendung der vorliegenden Programme ist die *Extended Symbolic Toolbox* erforderlich (vgl. Kap 4.2 u. 10) [19]. Damit stellt diese Programmsammlung keine vollständig eigenständige Ergänzung zur MATLAB-Grundversion dar.

Dagegen können die bei der Anwendung auszuführenden Optimierungen grundsätzlich mit der Funktion *fmins* aus der Grundversion durchgeführt werden (vgl. Kap. 3.5.1). Da MATLAB in der *Optimization Toolbox* über eine große Auswahl zusätzlicher sehr leistungsstarker Optimierverfahren verfügt, wird in der Programmsammlung die Option offen gehalten, gegebenenfalls diese Verfahren einzubinden.

Statistische Funktionen, die in dieser Arbeit eingesetzt werden und auch in der *Statistics Toolbox* vorhanden sind, werden separat zur Verfügung gestellt, um nicht auf diese Toolbox angewiesen zu sein.

3.3 Aufbau der Programmsammlung

Die entwickelten Programme sind analog zur Struktur der üblichen MATLAB-Erweiterungen nach Inhalt und Bezug in Verzeichnissen abgelegt. Jedes Verzeichnis ist mit Ausnahme der Zentralprogramme (vgl. Kap.3.5) thematisch in sich geschlossen. Allerdings können mehrere Verzeichnisse zum selben übergeordneten Themenkreis gehören. Aus Gründen der Übersichtlichkeit ist jedoch auf eine thematisch orientierte Verschachtelung verzichtet worden.
Die in Abb. 3-1. dargestellten Programmordner sind in Tab.3-1 den Kapiteln dieser Arbeit zugewiesen.

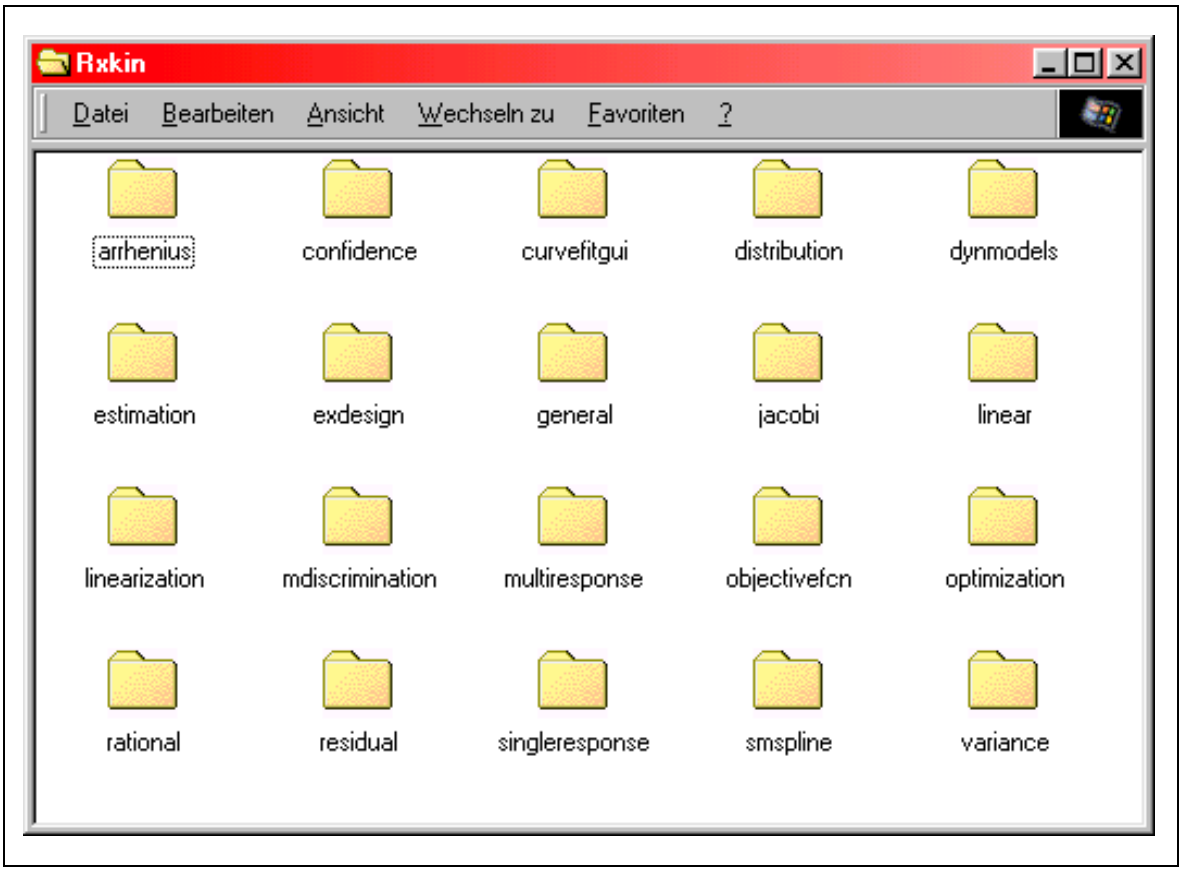

Abb. 3-1. Darstellung der Verzeichnisse der Programmsammlung unter WINDOWS

Tab. 3-1. Thematische Zuordnung der Verzeichnisse

Kapitel	Thema	Verzeichnis
3	Einführung in die Programmsammlung	*optimization, objectivefcn*
4	Grundlegende Programme	*general, distribution, jacobi*
5	Kurvenanpassung	*curvefitgui, rational, residual, smspline*
6	Lineare Regression in Einfachantwortsystemen	*linear, confidence*
7	Nichtlineare Regression in Einfachantwortsystemen	*linearization, singleresponse*
8	Regression in Mehrfachantwortsystemen	*multiresponse*
9	Modelldiskriminierung	*mdiscrimination*
10	Versuchsplanung	*exdesign*
11	Parameterschätzung in Differentialgleichungssystemen	*dynmodels*
12	Auswahl von Startwerten	*estimation*

Die Programmsammlung ist so angelegt, daß die Programme *matlab*-typische Funktionen darstellen und sich in das Konzept von MATLAB einfügen. Dabei kann formal zwischen Haupt- und Unterfunktionen unterschieden werden. Die Hauptfunktionen werden vom Anwender zur Problembearbeitung aufgerufen und koordinieren den Programmablauf sowie die dafür notwendigen Unterfunktionen. Die Unterfunktionen beinhalten notwendige oder häufig verwendete Programmteile eines Gesamtprogramms. Grundsätzlich kann eine Hauptfunktion auch als Unterfunktion bzw. Programmierelement eingesetzt werden. Wegen dieses Baukastenprinzips erfolgt die Problembearbeitung im allgemeinen durch die Kombination mehrerer Programme.

Bei der Bezeichnung der Funktionen dieser Programmsammlung ist jeweils darauf geachtet worden, daß sich aus der Abkürzung vor allem die Aufgabe des Programmpakets erschließen läßt. Außer bei allgemeinen Programmen, die vielseitig eingesetzt werden, repräsentieren die ersten zwei Buchstaben einer Funktionsbezeichnung das Verzeichnis, in dem sie sich befinden. Bei dem dritten und vierten Buchstaben handelt es sich gegebenenfalls um die rein inhaltliche Zuordnung zu einem Hauptprogramm, z. B. befindet sich *dmnumint* im Ordner *dynmodels* und *dmniopt* ist die zu *dmnumint* gehörende Optimierfunktion. Dieses ermöglicht bei alphabetischer Anordnung des Programmverzeichnisses

eines Ordners eine systematische Zuordnung der verschiedenen Funktionen. Durch diese Form der Bezeichnung soll lediglich die Zugehörigkeit zu einem Funktionenverband und damit zu einer Problemstellung deutlich sowie ein schneller Zugriff auf die Dateien ermöglicht werden. Eine Hervorhebung des hierarchischen Zusammenhangs der Funktionen ist nicht vorgenommen worden.

3.4 Vereinbarungen in der Benutzung – ybx-Schreibweise

Die Handhabung der vorliegenden Programme ist weitgehend analog zu der herkömmlichen Verwendung von MATLAB-Funktionen.
Jede erstellte Funktion verfügt in ihrem Programmkopf über detaillierte Informationen zu ihrem Verwendungszweck, Gebrauch und Aufruf, die im MATLAB-Hauptfenster über den *help*-Befehl abgerufen werden können.
Für den Einsatz einer Funktion ist die externe Vorgabe einer Argumentenliste notwendig. Um den Umfang dieser Argumentenliste gering zu halten, ist die MATLAB-Funktion *varargin* eingesetzt worden, wodurch die Belegung der Argumentenliste nicht bei jedem Aufruf vollständig erfolgen muß, sondern bei für den Programmablauf nicht unbedingt notwendigen Größen optional vorgenommen werden kann [15]. Dabei ist die Argumentenliste jeweils so angelegt worden, daß die einzelnen Argumente nach ihrer Bedeutung geordnet sind.

Wesentlicher Gegenstand der Programmsammlung ist die Parameterschätzung, die Beurteilung von Parametern und die Modellauswahl auf der Grundlage gegebener Modelle. Dazu muß die analytische Form der mathematischen Modelle bekannt sein.
An die Schreibweise und Vorgabe der Gleichungen werden gewisse Anforderungen geknüpft, um einen unverhältnismäßig hohen Arbeitsaufwand in der Programmierung zu umgehen. Die verwendete Schreibweise wird kurz als ybx-Schreibweise bezeichnet und orientiert sich sowohl an MATLAB-Schreibungen als auch an üblichen mathematischen Darstellungen. Bei der Formulierung der Modellgleichungen sind folgende Aspekte zu berücksichtigen:

1. Die rechten Seiten der Gleichungen, die zur Beschreibung eines Modells notwendig sind, werden einem *cell array* zugeordnet. Die einzelnen Gleichungen entsprechen dabei den Komponenten des *cell array* und werden jeweils als *string* deklariert.

2. Die Antworten werden als y bzw. bei Mehrfachantwortsystemen als y(1), y(2), ..., y(m) bezeichnet. Entsprechend werden die Parameter durch b und die unabhängigen Variablen durch x symbolisiert. Alle anderen Ausdrücke, wie z. B. Konstanten, werden durch ihren Zahlenwert in den Gleichungen repräsentiert.
3. Die Rechenoperatoren entsprechen den MATLAB-Schreibungen, wobei keine Matrizenoperationen verwendet werden dürfen, wie z. B. „`.*`" für die Komponentenmultiplikation von Matrizen inMATLAB.

Beispiel: Folgereaktion $A \xrightarrow{k_1} B \xrightarrow{k_2} C$

Modell: Mit $c_A = 1$ und $c_B = c_C = 0$ bei $t = 0$

$$c_A = \exp(-k_1 \cdot t)$$
$$c_B = \frac{k_1}{k_2 - k_1} \cdot \left(\exp(-k_1 \cdot t) - \exp(-k_2 \cdot t)\right) \qquad (3\text{-}1)$$
$$c_C = 1 - c_A - c_B$$

ybx-Schreibweise:

```
y(1)=exp(-b(1)*x);
y(2)=b(1)/(b(2)-b(1))*(exp(-b(1)*x)-exp(-b(2)*x));
y(3)=1-exp(-b(1)*x)-b(1)/(b(2)-b(1))*(exp(-b(1)*x)-exp(b(2)*x));
```

Formulierung des cell array:

```
ys = ...
 {'exp(-b(1)*x)'; ...
  'b(1)/(b(2)-b(1))*(exp(-b(1)*x)-exp(-b(2)*x))'; ...
  '1-exp(-b(1)*x)-b(1)/(b(2)-b(1))*(exp(-b(1)*x)-exp(b(2)*x))'};
```

3.5 Zentralprogramme und Erweiterungsmöglichkeiten

Bei den vorgestellten Programmen wird dem Benutzer prinzipiell die Möglichkeit eingeräumt, die vorhandenen Funktionen zur Parameterschätzung den problemspezifischen Bedürfnissen anzupassen oder zu erweitern. Dieses geschieht durch Zentralfunktionen bei der Verwaltung der Optimierverfahren und der Schätzfunktionen. Auf diese Zentralfunktionen kann relativ einfach zugegriffen und in ihnen können entsprechende Ergänzungen oder Änderungen vorgenommen werden, sofern die Ein- und Ausgabestrukturen dieser Funktionen eingehalten werden.

3.5.1 Optimierverfahren

Verzeichnis: *optimization*
Funktionen: *optrout.m*
optroutgrad.m

In MATLAB besteht eine übliche Optimierung aus einem Aufruf des Optimierverfahrens und einer Funktion, die vom Optimierverfahren selbst aufgerufen wird, da in ihr die Berechnung der Ziel- oder Schätzfunktion stattfindet, auf die optimiert wird. Um die Programme nicht fest an ein Verfahren zu binden oder einen direkten Eingriff in die Programme zu vermeiden, ist hier dem herkömmlichen, speziellen Optimierbefehl ein zentraler, allgemeiner Optimieraufruf vorgeschaltet. Dazu werden die speziellen Optimierbefehle in den Funktionen *optrout* bzw. *optroutgrad* verwaltet, deren Aufruf die Ausführung der gewählten Verfahren bewirkt. In diesen Routinen können Optimierbefehle von MATLAB oder externe Optimierverfahren eingebunden werden, sofern sie über die Struktur der MATLAB-Optimierungsaufrufe verfügen.
Durch die zentrale Organisation der Optimierverfahren wird zusätzlich erreicht, daß durch einen einmaligen Eingriff die Erweiterungen den meisten Programmen zur Verfügung gestellt werden und die Möglichkeit einer Verletzung der internen Programmstrukturen eingeschränkt wird.

Der Unterschied zwischen den Zentralfunktionen *optrout* und *optroutgrad* besteht darin, daß bei *optroutgrad* der Gradient auf die zu optimierende Funktion vorgegeben werden kann. Dazu wird der Gradient in einem speziellen *M-File* berechnet, und diese Datei wird über *optroutgrad* dem eigentlichen Optimierverfahren zugänglich gemacht [18].

Der Zusammenhang zwischen den Zentralfunktionen der Optimierung und den übrigen Programmen, die zu einem vollständigen Ablauf einer Schätzung erforderlich sind, ist in Abb. 3-2 dargestellt.
In der Programmstruktur wird von einer Hauptfunktion ausgegangen, in der die Ein- und Ausgabegrößen sowie die notwendigen Informationen gesammelt und gegebenenfalls vorab ausgewertet werden. Hier erfolgt stets der Aufruf von *optrout* oder *optroutgrad*. Nahezu alle Informationen werden an diese Routinen weitergegeben. Die Informationsübergabe erfolgt dabei entweder bei den für die Optimierverfahren notwendigen Größen als jeweils eigenständiges Argument oder gebündelt in einem *cell array*. Die Übermittlung als *cell array* erhöht die Überschaubarkeit der Programmierung und erweitert die Einsetzbarkeit der

Routinen, denn grundsätzlich können vor der vorletzten Stelle dieses Informationsfeldes beliebige Größen ergänzt werden (vgl. Kap. 3.1).

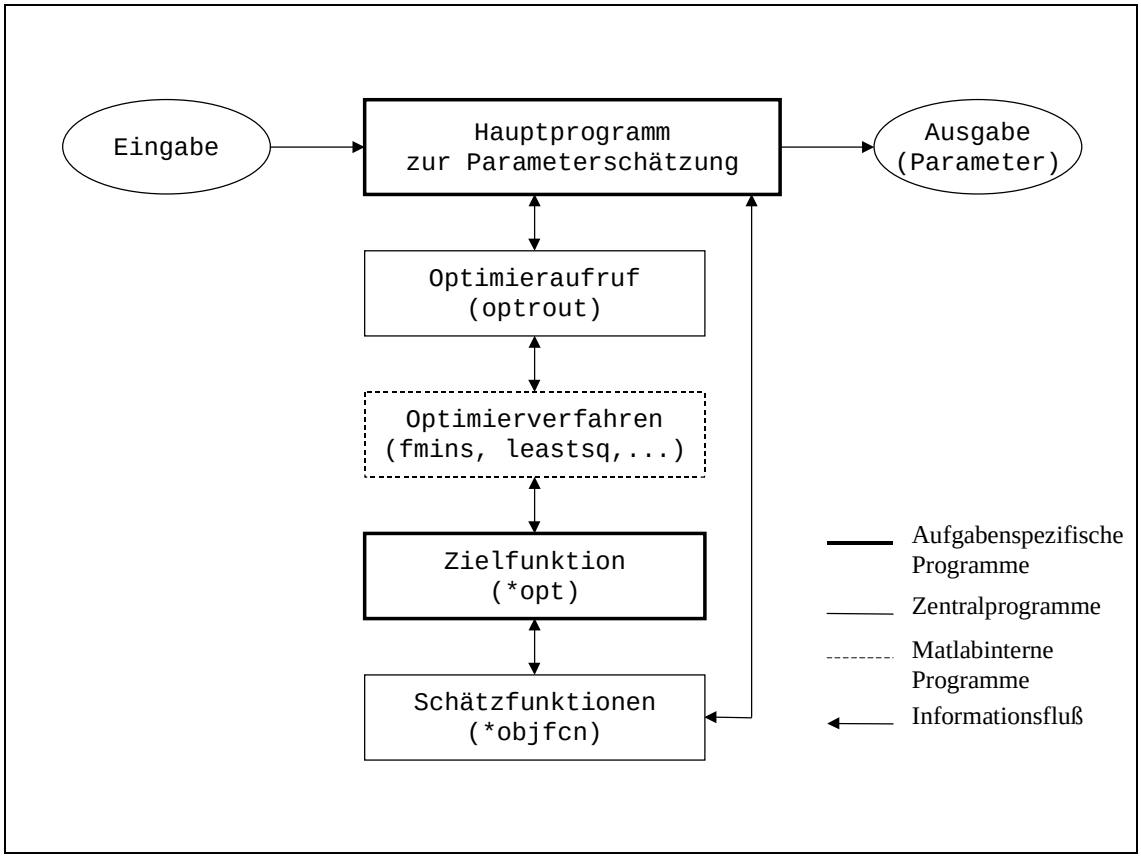

Abb. 3-2. Verknüpfung von problemspezifischen, *matlab*-internen und zentralen Programmabläufen

Vom Optimierverfahren selbst wird die Zielfunktion aufgerufen, in der die Modellantworten und Residuen für den aktuellen, vom Optimierverfahren vorgegebenen Parametersatz berechnet und an die zentral verwaltete Schätzfunktion weitergegeben werden. In Abhängigkeit der Residuen wird der Wert des Schätzkriteriums bestimmt und über die Zielfunktion an das Optimierverfahren übergeben, das auf dieser Grundlage entweder gezielt nach besseren Parametern sucht oder die Optimierung abbricht. Damit hat die Zielfunktion in Abb. 3-2 hauptsächlich die Aufgabe, ein spezielles Problem für ein allgemeines Kriterium zugänglich zu machen.
Um sich die Bewertungskriterien und andere bei der Schätzung entscheidende Größen ausgeben zu lassen, werden die Schätzroutinen im Hauptprogramm nochmals aufgerufen. Auf diese Weise wird auch der Einsatz globaler Variablen

eingeschränkt, die in Kombination mit anderen Programmen eine unnötige Fehlerquelle darstellen.
Ferner sind die vorliegenden Programme so angelegt, daß bei jeder Optimierung externe Veränderungen in den Einstellungen der Optimierverfahren, wie z.B. Schrittweite oder Toleranzbereiche, an das Hauptprogramm übergeben werden können. Dazu werden die Optimiereinstellungen in MATLAB einem Vektor zugewiesen, der dann lediglich in die entsprechende Position der Argumentenliste der Hauptfunktion eingesetzt wird [15].

Bei fast allen Optimierungsproblemen werden in der vorliegenden Programmsammlung standardmäßig drei Verfahren zur Auswahl gestellt. Bei den Optimierverfahren handelt es sich um *fmins*, *leastsq* und *constr* (vgl. Tab. 3-2).

Tab. 3-2. Eigenschaften einiger Optimierroutinen von MATLAB [18]

Aufruf	Verfahren	Beschreibung
fmins (Grundversion)	Simplex-Methode	Minimierung von unbeschränkten, mehrdimensionalen Funktionen; keine Berücksichtigung des Gradienten der Zielfunktion
leastsq (*Optimization Toolbox*)	Levenberg-Marquardt- oder Gauß-Newton-Verfahren	Lösung von mehrdimensionalen, nichtlinearen Schätzproblemen nach der Methode der kleinsten Fehlerquadrate; Bildung der Quadratsumme über *leastsq* selbst
constr (*Optimization Toolbox*)	SQP-Methode	Minimierung beschränkter, mehrdimensionaler Funktionen; Berücksichtigung von Nebenbedingungen

Die Funktion *fmins* ist Teil der MATLAB-Grundversion und kann ohne zusätzliche Ergänzungen eingesetzt werden, während *leastsq* und *constr* zur *Optimization Toolbox* gehören. Diese drei Funktionen sind so ausgewählt worden, daß sie die gängigen Problemstellungen abdecken (vgl. [7]).

Bei einer Optimierung mit *leastsq* kann optional ein Levenberg-Marquardt- oder eine Gauß-Newton-Verfahren eingesetzt werden, wobei die erste Methode standardmäßig bei *leastsq* voreingestellt ist.
Die Besonderheit einer *constr*-Optimierung besteht darin, daß eine Beschränkung des Optimierbereichs vorgenommen werden kann und Nebenbedingungen berücksichtigt werden können. Unter den Nebenbedingungen können Zusammenhänge zwischen den Parametern oder Antworten verstanden werden, wie z. B. stöchiometrische Beziehungen zwischen den Reaktionsteilnehmern bei der Untersuchung kinetischer Modelle. Dazu werden die einzelnen Nebenbedingungen in ybx-Schreibweise als *string* formuliert, in einem *cell array* abgelegt und den entsprechenden Programmen zugeführt.

3.5.2 Schätzfunktionen

Verzeichnis: *objectivefcn*
Funktion: *srobjfcn.m*
mrobjfcn.m
dmobjfcn.m

Die Schätzkriterien sind wie die Optimierverfahren ebenfalls in zentralen M-Dateien abgelegt. Diese befinden sich im Verzeichnis *objectivefcn*. In den Funktionsbezeichnungen steht dabei *sr* für Einfachantwortsysteme, *mr* für Mehrfachantwortsysteme und *dm* für dynamische Modelle bzw. Differentialgleichungssysteme.

Die wesentlichen Ausgabegrößen dieser Funktionen sind die Eingangsgröße für das Optimierverfahren, der Wert des Schätzkriteriums, die Ableitung der Schätzfunktion nach der Fehlermatrix (vgl. Kap. 8), die Varianz-Kovarianz-Matrix der Antworten und gegebenenfalls die Vorgabe von Nebenbedingungen (vgl. hierzu *constr* in [18]). Hierbei ist die Unterscheidung zwischen der Eingangsgröße für das Optimierverfahren und dem Wert des Schätzkriteriums sinnvoll, da diese Größen nicht notwendig gleich sein müssen. So greift beispielsweise *leastsq* auf den Residuenvektor zurück und berechnet programmintern den Wert der Fehlerquadratsumme als Schätzfunktion [18].
Die Eingabegrößen sind im allgemeinen der Residuenvektor oder die Matrix der Residuen, die Antwortmatrix, die Matrix der Modellantworten, Informationen über das gewählte Optimierverfahren und weitere für die Schätzfunktionen entscheidende Größen. Diese Argumentenliste kann vom Anwender relativ

einfach ergänzt werden, da auch hier die Möglichkeit der Deklaration einer variablen Argumentenliste über die MATLAB-Funktion *varargin* genutzt wird.

Die Programmstruktur dieser Zentralfunktionen gliedert sich in zwei Teile. Im ersten Teil wird über einen *switch*-Befehl die gewählte Schätzfunktion bearbeitet. Im zweiten Teil werden wiederum über einen Schalterbefehl die Schätzkriterien den oben bereits erwähnten Anforderungen und Möglichkeiten der Optimierverfahren angepaßt.

Auf die Details der berücksichtigten und eingebauten Schätzkriterien in den einzelnen Zentralfunktionen zur Schätzung wird in den entsprechenden Kapiteln eingegangen, in denen ihre Anwendung behandelt wird.

4 Grundlegende Programme

In diesem Abschnitt sollen wichtige Unterprogramme behandelt werden, die für die Anwendung der Programmsammlung nicht von zentraler Bedeutung sind, aber insgesamt zum Verständnis beitragen.

4.1 Auswertung von Modellgleichungen

Verzeichnis: *general*
Funktionen: *ycellval.m*

Mit der Funktion *ycellval* kann die Auswertung der rechten Seiten von Modellgleichungen nach Gl. (2-1) vorgenommen werden. Diese rechten Seiten müssen dazu in der ybx-Schreibweise formuliert und als *string* einem *cell array* zugeordnet werden (vgl. Kap. 3.4). Neben dem *cell array* der Modellgleichungen werden der Funktion *ycellval* die Matrix der unabhängigen Variablen und der Parametervektor vorgegeben. Die Modellgleichungen werden programmintern einer Stringverarbeitung unterzogen und die Modellwerte anschließend über die Anwendung des MATLAB-Befehls *eval* auf die Komponenten des *cell array* berechnet. Als Ausgabe erhält man eine Matrix der Modellantworten, in deren Zeilen die Antworten einer Modellgleichung für die verschiedenen Einstellungen der unabhängigen Variablen aufgeführt sind. (Vgl. Tab.4-1)

Tab. 4-1. Eigenschaften der Funktion *ycellval*

ycellval:	Auswertung der rechten Seiten von Modellgleichungen
Eingabe:	Modellgleichungen, Matrix der unabhängigen Variablen, Parametervektor
Ausgabe:	Matrix der Modellantworten

4.2 Die Jacobi-Matrix

Verzeichnis: *jacobi*
Funktionen: *jacsym.m,*
jacsyminv.m
jacsymval.m

Betrachtet man eine Funktion **f** gemäß Gl. (2-1), so ist die Ableitung dieser Vektorfunktion die sogenannte Funktional- oder Jacobi-Matrix **J** (Gl. (4-1)).

$$\mathbf{J} = \frac{d\mathbf{f}(\mathbf{x})}{d\mathbf{x}} \tag{4-1}$$

Die Komponenten der Jacobi-Matrix sind die partiellen Ableitungen der einzelnen Funktionen von **f** nach den Variablen. Dabei ist die Anordnung der Matrix nicht eindeutig festgelegt [20,21]. Deswegen wird im folgenden davon ausgegangen, daß sich in den Zeilen der Jacobi-Matrix die Ableitungen einer bestimmten Komponente von **f** nach den verschiedenen Variablen befinden.

Die Bedeutung der Jacobi-Matrix liegt bei der Parameterschätzung auf vier Gebieten:

- Linearisierung von funktionalen Zusammenhängen sowohl in den Parametern als auch in den unabhängigen Variablen, z.B. durch Taylor-Entwicklungen, wodurch unter anderem eine vereinfachte Parameterschätzung möglich wird (vgl. Kap. 7).
- Berechnung von Sensitivitätskoeffizienten durch Ableitung des Funktionenvektors nach den Parametern. Dabei entsprechen die Komponenten der Jacobi-Matrix den Sensitivitätskoeffizienten; sie sind ein Maß für die Empfindlichkeit, mit der sich die Parameter und die Antworten gegenseitig beeinflussen (vgl. Kap. 11).
- Vorstufe zur Aufstellung der Hesse-Matrix, die sich aus der Ableitung der Jacobi-Matrix ergibt und für die Ermittlung der Varianz-Kovarianz-Matrix einer Schätzung von Bedeutung ist (vgl. Kap. 7).
- Einsatz bei Integrationsroutinen und Optimierverfahren bei Berücksichtigung von Gradientenfunktionen, die über die Jacobi-Matrix berechnet werden (vgl. Kap. 11).

Die formale, analytische Jacobi-Matrix wird über die Funktion *jacsym* erhalten. Hierzu gibt man den Funktionenvektor gemäß der ybx-Schreibweise als *cell array* vor, wobei die einzelnen Funktionen separate Textelemente des *cell array* sind. Ferner muß die Dimension des Parameter- oder Variablenvektors angegeben werden. Das dritte Argument dieser Funktion ist ein *string*, der die Größe enthält, nach der abgeleitet wird. Über eine Stringverarbeitungsroutine werden diese Informationen so umgeformt, daß sie für die *Extended Symbolic Toolbox* lesbar sind [19]. Der Befehlsaufruf und die notwendigen, bearbeiteten Informationen werden als *string* formuliert und anschließend über den *eval*-Befehl ausgeführt. Man erhält jedoch von der *Extended Symbolic Toolbox* Ergebnisse in Strukturen, die für MATLAB nicht ohne weiteres verwertbar sind. Deshalb wird

das Ergebnis nochmals einer Stringverarbeitung unterzogen, wodurch wiederum ein *cell array* erhalten wird, das aus *strings* aufgebaut ist. Die Textelemente dieses Feldes entsprechen den partiellen Ableitungen in der Jacobi-Matrix und sind analog dazu in dem *cell array* angeordnet. Die Eigenschaften der Funktion *jacsym* sind in Tab. 4-2 zusammengefaßt.

Tab. 4-2. Eigenschaften der Funktionen zur Berechnung der Jacobi-Matrix

jacsym:	Erstellung der symbolischen Jacobi-Matrix als *cell array*
Eingabe:	Modellgleichungen, Dimension der Größe, nach der abgeleitet wird, Bezeichnung der Größe, nach der abgeleitet wird.
Ausgabe:	symbolische Jacobi-Matrix (Komponenten sind *strings* in einem *cell array*)
jacsyminv:	Erstellung der symbolischen inversen Jacobi-Matrix als *cell array*
Eingabe:	Modellgleichungen, Dimension der Größe, nach der abgeleitet wird, Bezeichnung der Größe, nach der abgeleitet wird.
Ausgabe:	symbolische inverse Jacobi-Matrix (s.o.)
jacsymval:	Auswertung der symbolischen Jacobi-Matrix aus *jacsym* / *jacsymval*
Eingabe:	Jacobi-Matrix gemäß *jacsym* / *jacsyminv*, Matrix der unabhängigen Variablen, Parametervektor
Ausgabe:	Jacobi-Matrix

Die Jacobi-Matrix für die verschiedenen Versuchspunkte wird mittels der Funktion *jacsymval* berechnet (vgl. Tab. 4.2). Dazu werden diesem Programm das über *jacsym* erhaltene *cell array*, die Parameterwerte und die Matrix der unabhängigen Variablen vorgegeben. Man erhält dann eine dreidimensionale Matrix, wobei in der dritten Dimension die Jacobi-Matrizen für die verschiedenen Versuchsniveaus aneinandergereiht sind.

Mit dem Programm *jacsyminv* sollte ursprünglich die inverse Jacobi-Matrix analog zu *jacsym* gebildet werden (vgl. Tab. 4.2). Dies ist jedoch nur eingeschränkt möglich, denn die Größe der Matrizen, die auf diese Weise invertiert werden können, ist von der Rechnerausstattung abhängig. Sie liegt bei einem Pentium 100 Rechner mit 32 MB Arbeitsspeicher bei 5x5-Matrizen. Hierbei handelt es sich um eine nur unter WINDOWS auftretende MATLAB-Fehlfunktion, die auch entsprechend im MATLAB-Hauptfenster angezeigt wird.

Als alternatives Verfahren zur Berechnung der inversen Matrix bietet sich daher an, die Jacobi-Matrix zunächst über *jacsym* aufzustellen, mit *jacsymval* zu evaluieren und anschließend zu invertieren, da dieses in MATLAB selbst durchgeführt werden kann und problemlos möglich ist.

Auf eine Besonderheit soll noch hingewiesen werden: Aufgrund der eingesetzten Stringverarbeitungstechniken ist es möglich, die Jacobi-Matrix auch für andere Gleichungen aufzustellen, die nicht der ybx-Schreibweise genügen. Wegen der Verknüpfung mit anderen Programmen wird hiervon im weiteren aber kein Gebrauch gemacht.

Die hier vorgestellten Funktionen haben den Vorteil, daß sie MATLAB-Programmen die analytische Form der Jacobi-Matrix zur Verfügung stellen können. Darin unterscheiden sie sich von einer in MATLAB 5.2 vorhandenen Funktion *numjac*, die aus den analytischen Gleichungen über ein Differenzenverfahren die Jacobi-Matrix ermittelt. Diese Funktion ist zwar nicht im Handbuch dokumentiert [15], aber über die *help*-Funktion in MATLAB zugänglich.
Beim Einsatz von *numjac* bei Optimierungs- oder Integrationsprozessen wird das verwendete Differenzenverfahren bei jedem Rechenschritt aufgerufen und durchgeführt. Daraus ergeben sich bei großen Systemen Verzögerungen in der Rechenzeit, was bei Verwendung von *jacsym* und *jacsymval* umgangen werden kann.

4.3 Verteilungsfunktionen

Verzeichnis: *distribution*
Funktionen: *normdistinv.m*
fdistinv.m
chi2distinv.m
tdistinv.m

Für die Durchführung bestimmter statistischer Auswertungsverfahren und die Berechnung unterschiedlicher Vertrauensbereiche ist die Kenntnis der Argumente verschiedener Verteilungsfunktionen von Bedeutung. Zu diesen Verteilungsfunktionen gehören:

- Normalverteilung,
- *F*-Verteilung,
- X^2-Verteilung,
- *t*-Verteilung.

Die dafür notwendigen Berechnungsgrundlagen sind zwar prinzipiell in der *Statistics Toolbox* vorhanden, werden hier aber zusätzlich eingebaut, um auf diese *Toolbox* nicht angewiesen zu sein (vgl. Kap. 3.2).

Für die Berechnung der Argumente der oben aufgeführten Verteilungsfunktionen stehen im Rahmen der Programmsammlung die in Tab.4-3 aufgeführten Funktionen zur Verfügung.

Tab. 4-3. Eigenschaften der Funktionen zur Argumentenberechnung von Verteilungsfunktionen

normdistinv:	Bestimmung des Arguments der normierten Normalverteilung bei vorgegebener Konfidenzzahl
Eingabe:	Konfidenzzahl
Ausgabe:	Argument der normierten Normalverteilung
fdistinv:	Bestimmung der Argumente für die *F*-Verteilung mit (*m*,*n*) Freiheitsgraden
Eingabe:	Signifikanzzahl, Freiheitsgrade
Ausgabe:	Argument der *F*-Verteilung
chi2distinv	Bestimmung der Argumente der X^2-Verteilung bei vorgegebener Konfidenzzahl
Eingabe:	Konfidenzzahl, Freiheitsgrade
Ausgabe:	Argument der X^2-Verteilung
tdistinv:	Bestimmung der Argumente der *t*-Verteilung bei vorgegebener Konfidenzzahl
Eingabe:	Konfidenzzahl, Freiheitsgrade
Ausgabe:	Argument der *t*-Verteilung

Bei der Erstellung dieser Funktionen sind zunächst mit Hilfe der *Statistics Toolbox* die Argumente für die entsprechenden Verteilungen in sinnvollen Bereichen berechnet und als Tabelle im Matrix-Format in jeweils einer entsprechenden MAT-Datei gespeichert worden. Die Richtigkeit der Tabellenwerte ist anhand von [10] überprüft worden. Die ...*distinv*-Funktionen laden bei ihrem Aufruf diese Tabellen und ermitteln aus ihnen den Argumentwert. Dazu wird gemäß der Vorgaben zwischen den Tabellenwerten linear interpoliert. Die notwendigen Vorgaben für die Verwendung der ...*distinv*-Funktionen sind eine Konfidenzzahl und die Anzahl der Freiheitsgrade (vgl.Tab. 4-3).

5 Kurvenanpassung

Unter einer Kurvenanpassung wird die Bestimmung mathematischer Modelle verstanden, die den Zusammenhang zwischen experimentellen Datensätzen wiedergeben [7]. Eine physikalische Grundlage für die Modelle ist dabei nicht entscheidend, da die mathematischen Zusammenhänge weitgehend zur Extra- und Interpolation sowie zur Beurteilung von Daten genutzt werden.
Im vorliegenden Fall werden nur Kurvenanpassungen für Modelle der Art

$$\eta = f(\boldsymbol{\beta}, x) \tag{5-1}$$

durchgeführt, die also eindimensional in den unabhängigen und abhängigen Variablen sind. Für diese Einschränkung der Modelle gibt es verschiedene Gründe: Zum einen werden in der Reaktionskinetik häufig Konzentrations-Zeit-Verläufe behandelt, zum anderen sollen die für die Anpassung entwickelten Programme vor allem zu Vorbetrachtungen dienen. Durch die erstellten Programme können damit erste Informationen zum Kurvenverlauf, zur Modellauswahl, zur Verteilungsfunktion sowie zu Parameterschätzwerten eingeholt oder integrale bzw. differentielle Datenverarbeitungen durchgeführt werden. Detailliertere Betrachtungen und umfangreichere Kurvenanpassungen können anschließend mit den speziellen Programmen zur Parameterschätzung gemacht werden, die in den nächsten Kapiteln näher erläutert werden.
Zentraler Gegenstand der Kurvenanpassung ist in dieser Programmsammlung ein *Graphical User Interface* (GUI), das mit den von MATLAB zur Verfügung gestellten Möglichkeiten entwickelt worden ist (vgl.Kap. 3.1). Auf Programme oder Verfahren, die sowohl in diesem GUI eingebunden sind als auch für sich genommen verwendet werden können, wird gesondert in diesem Kapitel eingegangen.

5.1 GUI zur Kurvenanpassung

Verzeichnis: *curvefitgui*
Funktionen: *curvefitgui.m*

Das zur Kurvenanpassung erstellte GUI wird über den Aufruf *curvefitgui* aktiviert. Das Hauptfenster ist in Abb. 5-1 dargestellt.

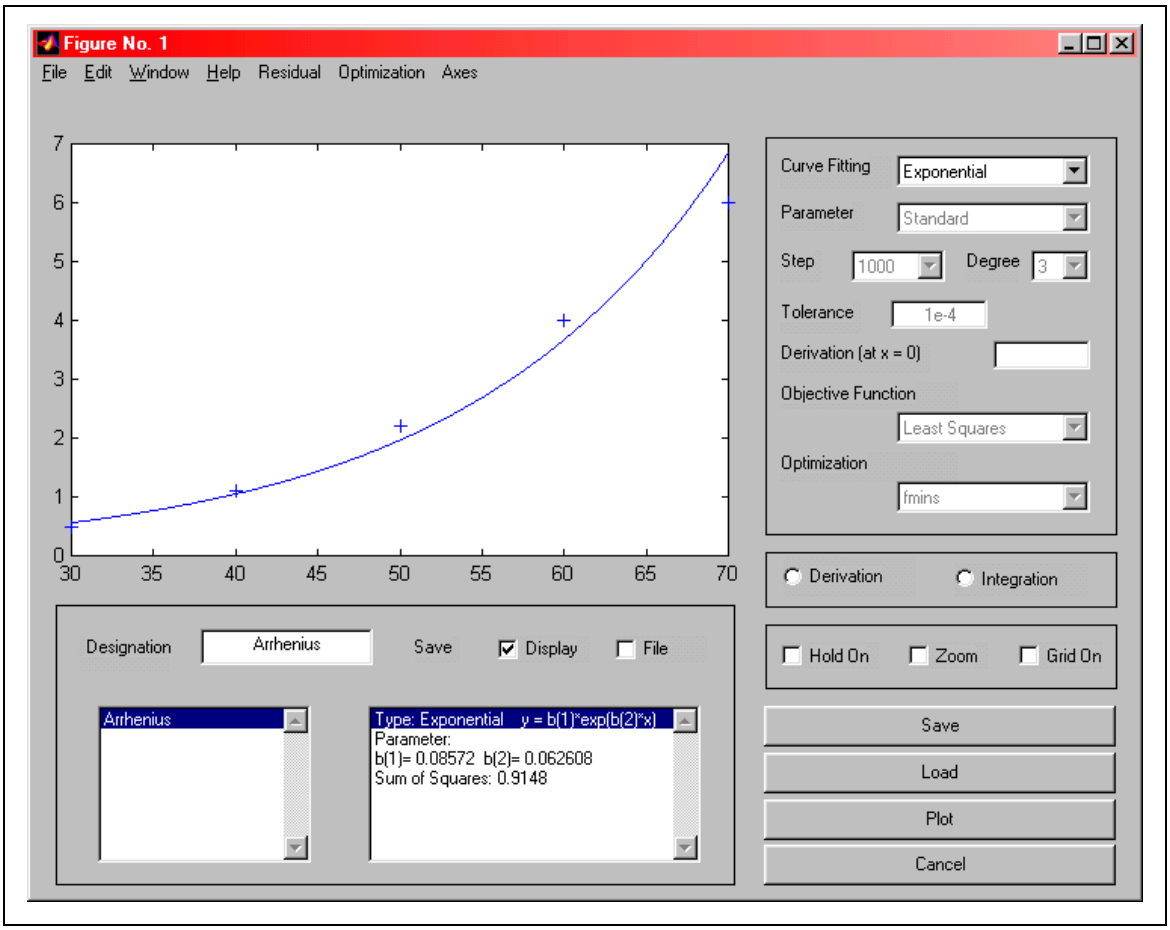

Abb. 5-1. GUI-Oberfläche zur Kurvenanpassung

5.1.1 Beschreibung des GUI

Laden von Daten

Beim Gebrauch dieses GUI müssen zunächst die entsprechenden Datensätze geladen werden. Dieses geschieht durch die Betätigung des Schalters (*pushbutton*) *Load*, durch den ein zweites Fenster aufgerufen wird. Beim Einlesen besteht die Option, die Daten aus ASCII-, M- oder MAT-Dateien zu laden. Anschließend können die entsprechenden Variablen den Achsen des Diagramms im Hauptfenster zugeordnet werden. Die Eingaben müssen über den *OK*-Schalter in dem entsprechenden Fenster zum Dateneinlesen bestätigt werden.

Modellauswahl

Für die eigentliche Kurvenanpassung über die Hauptoberfläche kann aus der Auswahlliste (*popupmenu*) *Curve Fitting* zwischen verschiedenen mathematischen Modellen gewählt werden. Diese Modelle umfassen sowohl über-

arbeitete MATLAB-Funktionen als auch zusätzliche Funktionen. Dabei handelt es sich um die Wahlmöglichkeiten:

None, Linear, Polynomial, Rational, Exponential, Spline, Smooth. Spline, Userdefined.

None bezieht sich hierbei auf die graphische Auftragung der Daten ohne Anpassung. Die Anpassung an lineare (*Linear*), ganzrationale (*Polynomial*) und exponentielle (*Exponential*) Modelle erfolgt intern über die entsprechenden MATLAB-Befehle, wobei die Parameter zur exponentiellen Darstellung über eine Logarithmierung der Exponentialfunktion bestimmt werden. Für die ganzrationalen und rationalen Funktionen wird der Polynomgrad über eine Auswahlliste bzw. zwei Auswahllisten *Degree* festgelegt.
Bei der Verwendung von Splinefunktionen wird unter *Spline* auf Elemente der MATLAB-Grundversion zurückgegriffen, wobei es sich um interpolierende, polynomiale Splines dritten Grades handelt. Die Ausgleichssplines (*Smooth. Spline*) werden ebenso wie die rationalen Funktionen in den nachfolgenden Abschnitten erläutert.
Bei Kurvenanpassungen an benutzerdefinierte Modelle (*Userdefined*) muß die Modellgleichung in einem gesonderten Fenster in der ybx-Schreibweise vorgegeben werden (vgl. Kap. 3.4). Dieses Fenster wird automatisch bei Wahl der entsprechenden Option aufgerufen.

Durchführung der Kurvenanpassung
Die Art der Kurvenanpassung kann bei einigen Modellen mit den standardmäßigen Voreinstellungen oder benutzerdefiniert erfolgen, indem in der Auswahlliste *Parameter* die dazugehörige Option gewählt wird. Hierdurch können teilweise zusätzliche Einstellmöglichkeiten freigeschaltet werden. Dazu gehören die Möglichkeit der Vorgabe einer Schrittweite bei *Step*, über die eine relativ glatte Kurvendarstellung durch Erhöhung der Anzahl der intern berechneten Modellwerte erreicht wird. Ferner kann über *Tolerance* festgelegt werden, wie groß Parameter mindestens sein müssen, um als von Null verschieden betrachtet und damit in der Kurvendarstellung berücksichtigt zu werden.
Die Modellfunktionen unterscheiden sich jedoch nicht nur in ihrer mathematischen Formulierung, sondern auch in der Art der Programmierung. Im Gegensatz zu den übrigen Funktionen erfolgt die Parameterschätzung in den gebrochen-rationalen und benutzerdefinierten Modellen nicht mehr *matlab*-intern, sondern über den Einsatz von Optimierverfahren. Deswegen besteht bei diesen Funktionen die Möglichkeit, die Schätzfunktion (*Objective Function*) und

das Optimierverfahren (*Optimization*) zu wählen. Ferner können zusätzliche Informationen über den Ableitungswert an der Stelle Null der Abszisse bei der Schätzung berücksichtigt werden (*Derivation (at x=0)*).
Bei den Schätzfunktionen wird zwischen der ungewichteten, der gewichteten und der benutzerdefiniert gewichteten Fehlerquadratsumme unterschieden. Bei der ungewichteten Fehlerquadratsumme werden Wiederholungsmessungen nur insofern berücksichtigt, als für die Wiederholungsmessungen ein Mittelwert gebildet wird, der ungewichtet in die Quadratsumme einfließt. Dagegen wird im gewichteten Fall die Anzahl der Wiederholungsmessungen pro Versuchspunkt beachtet. Bei Verwendung der benutzerdefiniert gewichteten Schätzfunktion wird ein weiteres Fenster aufgerufen, über das die Gewichte für die Beobachtungswerte eingetragen werden können. Dabei kann optional auch die Eintragung desselben Gewichts für alle Werte vorgenommen werden, um so die Größenordnung der Residuenquadratsumme zu beeinflussen und damit gegebenenfalls die Schätzung zu vereinfachen.
Die Auswahl der Optimierverfahren kann zwischen *fmins*, *leastsq* und *constr* getroffen werden. Dazu werden bei den benutzerdefinierten Funktionen in dem Fenster, in dem auch die Modellgleichung formuliert wird, die Startschätzwerte vorgegeben. Bei Verwendung von *constr* können hier ebenfalls Parameterbereichsgrenzen oder Nebenbedingungen eingetragen werden (vgl. Kap. 3.5.1).
Die Kurvenanpassung und Parameterschätzung mit *Userdefined* ist eine Vorstufe der Parameterschätzung in Einfachantwortsystemen mit der Funktion *singleres* (Kap. 7.3).
Neben der Wahl des Optimierverfahrens können auch die wichtigsten Optimiereinstellungen über die Menüleiste unter *Optimization* verändert werden.
Eine Aktualisierung des Graphen entsprechend der Einstellungen zur Kurvenberechnung erfolgt erst durch die Betätigung des *Plot*-Schalters.

Differentiation und Integration
Die eingelesenen Daten können durch Setzen der entsprechenden Optionen (*radiobutton*) numerisch integriert oder differenziert werden. Die Integration erfolgt gemäß einer Trapeznäherung unter Berücksichtigung auch nichtäquidistanter Integrationsstellen nach [22].
Die Differentiationsverfahren sind dagegen abhängig vom ausgewählten Kurvenmodell. Soweit es möglich ist, wird hierbei intern die Ableitungskurve analytisch aufgestellt.
Bei den Splines wird die Differentiation durch Überführung des Differentialquotienten in einen Differenzenquotienten durchgeführt, was wegen der hohen

Ungenauigkeit dieses Verfahrens nur bei einer großen Anzahl von Funktionswerten ratsam ist [5].
Bei den rationalen und benutzerdefinierten Funktionen sowie bei der unangepaßten Auftragung der Daten erfolgt das Differenzieren numerisch über eine Polynomnäherung an der Differentiationsstelle.
Bei der Differentiation und Integration unter der Einstellung *None* werden die entsprechenden Werte an den Meßstellen ermittelt, was beispielsweise für die Ermittlung von Reaktionsgeschwindigkeiten von Bedeutung ist.

Datenspeicherung
Die Einstellungen einer Kurvenanpassung, wie z.B. Kurventyp, Parameterschätzwerte, Optimiereinstellungen und Wert des Schätzkriteriums, werden stets in die rechte *listbox* eingetragen (vgl. Abb. 5-1). Für diese Informationen und die ermittelten Werte (Modellwerte, Ableitungen etc.) gibt es zwei mögliche Speicheroptionen.
Zum einen können zum Vergleich von auf verschiedene Weise angepaßten Kurvenverläufen die Einstellungen im Hauptfenster durch „Anklicken" des Kontrollfeldes (*checkbox*) *Display* gespeichert werden. Hierfür kann zur aktuellen Kurvenbetrachtung im Eingabebereich (*edit*) *Designation* eine Kurvenbezeichnung eingegeben werden. Unter diesem Titel, der in die linke *Listbox* eingetragen wird, werden alle dazugehörigen Daten gespeichert. Bei mehreren Speicherungen werden durch „Anklicken" des gewünschten Titels die entsprechenden Einstellungen in der rechten *Listbox* aufgerufen und der Kurvenverlauf im Graphikbereich dargestellt (vgl. Abb. 5-1).
Dagegen werden zum anderen bei Aktivierung der *checkbox Extern* unter der eingegebenen Bezeichnung zwei externe Dateien angelegt. Die numerischen Werte werden in einer ASCII-Datei und die Einstellungen bei der Kurvenanpassung als *cell array*-Variable *info* in einer MAT-Datei abgelegt. Diese Dateien können weiteren Anwendungen zugeführt werden.

Bearbeitung der graphischen Darstellung
Die *matlab*-üblichen Optionen in der graphischen Darstellung, wie z.B. Ausschnittsvergrößerung (*Zoom*), gleichzeitige Betrachtung mehrerer Kurven (*Hold On*) oder Einfügen eines Gitternetzes (*Grid On*), können über die vorgegebenen Kontrollfelder aktiviert werden.
Eine halb- oder doppeltlogarithmische Auftragung der Kurvenverläufe ist über die Menüleiste unter *Axes* möglich. Dabei besteht die Wahl zwischen dem natürlichen und dekadischen Logarithmus.

Fehleranalyse
Für eine erste Analyse zur Richtigkeit des gewählten Modells oder zur Kontrolle, ob eine Normalverteilung der Fehler vorliegt, stehen in der Menüleiste unter *Residual* zwei Verfahren zur Verfügung. Zum einen können mit der Wahl der Option *Plot* die Residuen zu den entsprechenden Meßstellen aufgetragen werden, zum anderen kann über *Distribution* die Verteilung der Residuen betrachtet werden. Bei dem letzten Verfahren wird die Anzahl der Residuen über die normierten Residuen aufgetragen und je nach Vorgabe eine Vertrauensgrenze bei zugrunde gelegter Normalverteilung eingezeichnet.
Beide Methoden sind unabhängig von diesem GUI einsetzbar und werden deshalb in Kap. 5.4 näher beschrieben.

5.1.2 Umsetzung in MATLAB und Beurteilung

GUIs werden in MATLAB über den Aufruf *guide* erstellt. Dabei wird die Gestaltung des GUIs und das Einfügen von visualisierten Programmierbausteinen, wie Schaltern (*pushbuttons*), Auswahllisten (*popupmenu*), Kontrollfeldern (*checkbox*), über eine *windows*-ähnliche Kontrolloberfläche verwaltet. Über diese Kontrolloberfläche sind weitere Oberflächen zugänglich, die die optische Anordnung, die Eigenschaften und den Befehlsinhalt der Programmierelemente oder den Aufbau der Menüleiste regeln. Die Zuweisung einer Befehlsabfolge zu einem Programmbaustein erfolgt hierbei über den *Callback Editor*. In der Umsetzung gibt es prinzipiell mehrere Möglichkeiten. Zum einem kann die Befehlsabfolge als Eigenschaft dem Programmelement zugeordnet werden, was zu längeren Rechenzeiten führen kann [14]. Zum anderen kann der Programmteil in einer Funktion abgelegt werden, die dem Programmelement zugewiesen wird. Diese Variante ist nach [14] in ihren Eigenschaften robuster, schneller und leichter programmierbar. Letztere Eigenschaft läßt sich auch im Rahmen dieser Arbeit bestätigen, da auf diese Weise die Programmteile ohne Umwege über die GUI-Programmieroberflächen im MATLAB-Editor bearbeitet und kontrolliert werden können.

Bei der Anwendung von *curvefitgui* sind dem zentralen GUI sieben weitere GUIs untergeordnet, die Aufgaben bezüglich des Ladens von Daten oder der Eingabe von Modellen übernehmen.
Aufgrund der *callback*-Programmierung über Funktionen besteht das gesamte Konzept aus 71 Objekten, die größtenteils M-Dateien und wenige MAT-Dateien sind.

Die eingebundenen Funktionen und Verknüpfungen sind weitgehend speziell für das GUI angelegt worden, so daß Veränderungen in anderen Teilen der Programmsammlung keinen Einfluß auf das GUI haben. Ausnahmen sind lediglich die zentrale Verwaltung des Optimieraufrufs durch *optrout*, die Funktionen zur Berechnung der Jacobi-Matrix und die Programme zur Residuenanalyse (vgl. Kap. 3.5.1, 4.2 u. 5.4).

Die Erstellung von GUIs in MATLAB ist aus einer Erweiterung von P. MARCHAND zur Version MATLAB 4.0 entstanden [23], die seit der Version 5.0 fester Bestandteil der MATLAB-Grundversion ist. Die Funktionsfähigkeit der MATLAB-Elemente zur GUI-Erstellung hat sich mit höheren Versionen verbessert und ist absturzsicherer geworden. Dennoch sind einige Probleme aufzuzeigen. Zum einen ist die vollständige Akzeptanz der GUIs früherer Versionen gegenüber höheren eingeschränkt. Dieses trifft vor allem auf die Version 5.2 zu, die trotz entsprechender Erweiterung auf 5.2.1 *frame*-Darstellungen früherer Versionen nicht ohne weiteres umsetzen kann. Ferner können nicht alle GUI-Eigenschaften problemlos besetzt werden. Beispielsweise ist es nicht möglich, Oberflächenbezeichnungen im *Property Editor* unter *Name* in MATLAB 5.2.1 zu setzen.

Insgesamt läßt sich aufgrund der Erfahrungen bei der Entwicklung dieser Programmsammlung feststellen, daß der Arbeitsaufwand bei der Erstellung umfangreicher GUIs, wie im vorliegenden Fall, im Verhältnis zur Programmierung von MATLAB-Funktionen unangemessen hoch ist. Ferner ist die Überschaubarkeit der Programmstrukturen durch die Vielzahl der miteinander verknüpften Objekte eingeschränkt und die Durchführung von Veränderungen im Programminhalt wegen des Zusammenspiels der Routinen erschwert.
Der Vorteil der GUIs besteht dagegen vor allem in der Bedienungsfreundlichkeit, denn durch die Schaltoberfläche ist die Zuweisung von Argumenten zu Funktionen und ihre Ausführung einfacher. Zusätzlich ist es auch als vorteilhaft zu bewerten, daß MATLAB grundsätzlich diese Programmiertechnik zur Verfügung stellt.

5.2 Rationale Funktionen

Verzeichnis: *rational*
Funktionen: *ratiofit.m*
ratioval.m

MATLAB verfügt standardmäßig nicht über Funktionen zur Berechnung von Koeffizienten gebrochen-rationaler Funktionen gemäß Gl.(5-2) und der Bestimmung ihrer Funktionswerte.

$$\eta = \frac{\sum_{i=0}^{n} a_i \cdot x^i}{\sum_{j=0}^{m} b_j \cdot x^j}. \qquad (5\text{-}2)$$

Diese Funktionen haben jedoch für die Darstellung von Konzentrations-Zeit-Verläufen bei Reaktionsnetzwerken oder heterogen katalysierten Reaktionen eine erhebliche Bedeutung [24,25], weshalb für die Kurvenanpassung entsprechende Programme entwickelt worden sind .
Die Schätzung der Koeffizienten gebrochen-rationaler Funktionen erfolgt über die üblichen Optimierverfahren (*fmins, leastsq, constr*) mit der Routine *ratiofit* (vgl. Kap. 5.1). Als Schätzfunktionen können die unter Kap. 5.1.1 beschriebenen Kriterien eingesetzt werden, die mit der Funktion *residual* in der Unterfunktion *ratioopt* berechnet werden (vgl. Kap. 5.4). Die Vorgaben für die Anwendung von *ratiofit* sind im wesentlichen die unabhängigen und abhängigen Größen sowie jeweils der Grad vom Zähler- und Nennerpolynom. Die Berechnung der Funktionswerte kann analog zu *polyval* mit der Funktion *ratioval* erfolgen [15].
Die Eigenschaften von *ratiofit* und *ratioval* sind in Tab. 5-1 aufgeführt.

Tab. 5-1. Eigenschaften der Funktionen zur Berechnung rationaler Funktionen

ratiofit:	Bestimmung der Koeffizienten in rationalen Funktionen
Eingabe:	Vektor der unabhängigen und abhängigen Variablen, Polynomgrad von Zähler und Nenner, Optimieroptionen*, Optimierverfahren*, Schätzfunktion*, Gewichtsvektor*
Ausgabe:	Polynomkoeffizienten von Zähler und Nenner
ratioval:	Berechnung der Funktionswerte rationaler Funktionen
Eingabe:	Polynomkoeffizienten von Zähler und Nenner, Vektor der unabhängigen Variablen
Ausgabe:	Vektor der Funktionswerte

* = Eingabe freigestellt

5.3 Ausgleichssplines

Verzeichnis: *smspline*
Funktionen: *smspline.m*

Bei Splinefunktionen handelt es sich im allgemeinen um Funktionen, die sich abschnittsweise aus Polynomen zusammensetzen und zur Interpolation zwischen Datenpunkten und zur Erzeugung glatter Kurven dienen [26]. Die von herkömmlichen Splinefunktionen erzeugten Kurven verlaufen dabei durch die betrachteten Datenpunkte. In der MATLAB-Grundversion kann man interpolierende Polynomsplines dritten Grades über die Funktion *spline* erstellen [15].
Über die *Spline Toolbox* ist es ferner möglich, ausgleichende Splines zu behandeln, die aufgrund eines eingeschränkten Vertrauens in die Daten in ihrer Kurvendarstellung nicht notwendig durch die Meßpunkte verlaufen. Um diese Möglichkeiten auch bei der hier vorgestellten Kurvenanpassung zu nutzen, ist die Funktion *smspline* erstellt worden, die auf einen Algorithmus nach [26] zurückgreift. Über *smspline* werden Ausgleichssplines über Polynome dritten Grades erstellt, wobei die Randableitungen erster Ordnung der Kurven berücksichtigt werden können. Damit vereint diese Funktion die Eigenschaften der Funktionen *csape* und *csaps* aus der *Spline Toolbox* [27].
Die grundlegenden Eigenschaften von *smspline* befinden sich in Tab. 5-2.

Tab. 5-2. Eigenschaften der Funktion *smspline*

smspline:	Berechnung polynomialer Ausgleichssplines dritten Grades mit vorgegebener erster Randableitung
Eingabe:	Vektor der unabhängigen und abhängigen Variablen, erste Randableitung links, erste Randableitung rechts, Gewichtsvektor*, Anzahl der Punkte zwischen den Meßstellen (zur besseren graphischen Darstellung)*
Ausgabe:	Funktionswerte der Meßstellen, Funktionswerte an den zusätzlichen Stellen

* = Eingabe freigestellt

Der Programmablauf gliedert sich in zwei Funktionen, von denen *smspline* als Hauptfunktion die Ein- und Ausgabe sowie die Berechnung der Daten für den Kurvenverlauf regelt (vgl. Tab. 5-2.). Die eigentliche Bestimmung der Polynomkoeffizienten wird intern über den Aufruf der Funktion *regspl1d* durchgeführt, in der auch der Algorithmus abgelegt ist. Bei der Umsetzung dieses Berechnungsverfahrens hat sich vor allem wegen der Vielzahl der auftretenden Polynome die Matrizenverarbeitung von MATLAB bewährt.
Die Festlegung der Abweichung des Kurvenverlaufs von den tatsächlichen Daten erfolgt in *smspline* über die Vorgabe eines Gewichtsvektors. Dabei sollten die Gewichte zwischen 0 und 1 liegen, wobei 1 für uneingeschränktes Vertrauen in

die Daten steht. Die Gewichte können auf die einzelnen Meßpunkte bezogen in Vektorform oder pauschal als Skalar vorgegeben werden.

5.4 Residuenanalyse

Verzeichnis: *residual*
Funktionen: *residual.m*
nresonum.m

Die im Kurvenanpassungs-GUI zur Verfügung gestellten Methoden zur Residuenanalyse können unabhängig vom Gebrauch des GUI eingesetzt werden. Die im folgenden vorgestellten Verfahren stellen graphische Beurteilungsmöglichkeiten getroffener Modellannahmen bei der Parameterschätzung dar und sind [8] entnommen. Bei den zu überprüfenden Annahmen handelt es sich im wesentlichen um die Forderung nach Unabhängigkeit und Normalverteilung der Fehler, wobei für die Normalverteilung der Mittelwert Null und eine konstante Varianz angenommen wird. Die Methoden erweitern die im MATLAB-Handbuch [15] vorgestellten Vorgehensweisen und setzen im Gegensatz zur *Statistics Toolbox* andere Schwerpunkte [17].
Mit der Funktion *residual* erfolgt die Berechnung der Residuen. Werden bei ihrem Aufruf nur die Modellwerte und Beobachtungswerte vorgegeben, erfolgt die uneingeschränkte Residuenberechnung, wodurch vorhandene Wiederholungsmessungen entsprechend mehrfach eingehen. Fügt man dagegen der Argumentenliste den Vektor der unabhängigen Variablen hinzu, werden die Residuen an derselben Meßstelle gemittelt. Die erhaltenen Mittelwerte gehen ohne Berücksichtigung der Vielfachheit in die Ausgabe des Residuenvektors ein.
Da diese Funktion auch bei den Schätzverfahren eingesetzt wird, liegt noch eine weitere Option auf Gewichtung der Residuen vor (vgl. Kap.5.1 u. Tab. 5-3).

Tab. 5-3. Eigenschaften der Funktionen zur Residuenanalyse

residual:	Gewichtete und ungewichtete Residuenberechnung
Eingabe:	Beobachtungswerte, Modellwerte, Werte der unabhängigen Variablen*, Gewichtsvektor*
Ausgabe:	Residuen
nresonum:	Auftragung der normierten Residuen über der Zahlengerade
Eingabe:	Residuen, Parameteranzahl, Vertrauensgrenzen
Ausgabe:	Anzahl der Residuen pro Zahl, dazugehörige Zahlenwerte, graphische Darstellung

* = Eingabe freigestellt

Die mit *residual* erhaltenen Residuen können graphisch über ihre zeitliche Aufnahmereihenfolge aufgetragen werden. Ist das so erhaltene zeitliche Band zu einer Seite gespreizt, läßt sich eine nicht konstante Fehlervarianz vermuten. Eine Steigung oder Krümmung im Residuenband legt dagegen den Schluß nahe, daß eine zeitliche Abhängigkeit in der Regressionsgleichung vernachlässigt worden ist.

Bei der zweiten Methode wird die Funktion *nresonum* verwendet. Mit ihrer Hilfe wird das mittlere Residuenquadrat als Schätzung der Fehlervarianz berechnet. Anschließend werden die Residuen darauf normiert und auf der Zahlengerade angetragen. Bei Erfüllung der oben genannten Voraussetzungen müssen die Residuen dabei einer $N(0,1)$-verteilten Funktion genügen. Die extern vorgegebenen Vertrauensgrenzen werden ebenfalls in die graphische Darstellung eingebunden. Für die dafür notwendigen Berechnungen wird *normdistinv* verwendet (vgl. Kap. 4.3). Liegen unter den getroffenen Voraussetzungen mehr Residuen als statistisch zulässig außerhalb des Vertrauensbereichs, müssen die Annahmen überprüft werden.

6 Lineare Regression bei Einfachantwortsystemen

Regressionsverfahren können unterschiedlich klassifiziert werden. Es kann beispielsweise zwischen linearer und nichtlinearer Regression oder zwischen Regressionen bei Ein- und Mehrfachantwortsystemen unterschieden werden [28].
Im Rahmen dieser Arbeit erfolgt die Einteilung zwischen Ein- und Mehrfachantwortsystemen, wobei nur bei Einfachantwortsystemen eine Unterscheidung zwischen linearer und nichtlinearer Regression vorgenommen wird.
Obwohl die theoretischen Grundlagen der Einfachantwortsysteme als Spezialfall der Theorie für Mehrfachantwortsysteme betrachtet werden können, spricht folgendes für eine getrennte Behandlung [9]:

- Einfachantwortsysteme sind teilweise experimentell leichter zugänglich und können deshalb möglicherweise detaillierter untersucht werden, z. B. bezüglich statistischer Größen wie der Varianz der Antworten.
- Die Schätzfunktionen bei Mehrfachantwortsystemen sind umfangreicher, da sie gegebenenfalls den Einfluß der verschiedenen Antworten berücksichtigen müssen, was bei Anwendung dieser Kriterien auf Einfachantwortsysteme unter anderem zu Rundungsfehlern führen kann.
- Bei der Behandlung von Mehrfachantwortsystemen sind gegenüber Einfachantwortsystemen Abhängigkeiten in den Antworten oder den Erwartungswerten der Antworten zu berücksichtigen.

Auch die separate Behandlung der linearen und nichtlinearen Regression bei Einfachantwortsystemen ist sinnvoll, da

- MATLAB für lineare Regressionen spezielle Lösungsverfahren zur Verfügung stellt und
- lineare Systeme aufgrund ihrer Interpolationsfunktion andere Untersuchungsschwerpunkte haben, wie z. B. die Genauigkeit, mit der die abhängige Variable wiedergegeben werden kann.

In den nachfolgenden Abschnitten werden Methoden zur Parameterschätzung in linearen Gleichungssystemen und statistische Testverfahren zur Beurteilung von Regressionsergebnissen vorgestellt. Dabei sind die statistischen Verfahren in ihrem Einsatz nicht auf die lineare Regression begrenzt.

6.1 Parameterschätzung in linearen Gleichungssystemen

Verzeichnis: *linear*
Funktionen: *linreg.m*
mlinreg.m

Unter einer linearen Regression versteht man die Parameterschätzung in Modellen, die linear in den Parametern sind. Diese Modelle lassen sich mathematisch folgendermaßen beschreiben [8,29]

$$\eta = \beta_0 + \sum_{i=1}^{m} \left(\beta_i \cdot x_i \right) = \mathbf{X} \cdot \boldsymbol{\beta} \tag{6-1}$$

Neben linearen Gleichungssystemen können auch nichtlineare Systeme einer linearen Regression unterzogen werden, wenn sie sich auf die Struktur von Gl. (6-1) transformieren lassen. Diese Modelle werden dann als „eigentlich linear" bezeichnet [9]. Dazu gehören unter anderem Polynome oder die logarithmierte Arrhenius-Gleichung. Da derartige Transformationen zur Fehlerverzerrung führen können, ist die Gültigkeit der Durchführung der linearen Regression, z. B. durch die Residuenanalyse, zu überprüfen (vgl. Kap. 5.4).
Die Bedeutung der durch Gl. (6-1) beschriebenen Systeme reicht von der Aufstellung von Kalibriergeraden bis zur Auswertung von Faktorplänen höheren Grades [30,31].

Für die Parameterschätzung durch lineare Regression gibt es zwei grundsätzliche Vorgehensweisen: die schrittweise lineare Regression und die simultane multiple Parameterschätzung. Da das erste Verfahren in der *Statistics Toolbox* umgesetzt ist [17], wird in dieser Programmsammlung lediglich die simultane Parameterschätzung behandelt.
Die zu Gl. (6-1) gehörende Regressionsgleichung lautet dementsprechend

$$\hat{\mathbf{y}} = \mathbf{X} \cdot \mathbf{b} = \begin{pmatrix} 1 & x_{11} & \cdots & x_{1m} \\ \vdots & \vdots & \cdots & \vdots \\ 1 & x_{n1} & \cdots & x_{nm} \end{pmatrix} \cdot \begin{pmatrix} b_0 \\ \vdots \\ b_m \end{pmatrix} \tag{6-2}$$

Die Lösung von Gl. (6-2) ergibt sich gemäß der Methode der kleinsten Fehlerquadrate nach [22] zu

$$\mathbf{b} = \left(\mathbf{X}^T \cdot \mathbf{X} \right)^{-1} \cdot \mathbf{X}^T \cdot \mathbf{y} \tag{6-3}$$

Bei Berücksichtigung mehrerer unabhängiger Variablen und einer großen Anzahl an Versuchen kann ein Vorgehen gemäß Gl. (6-3) unter anderem aufgrund von Rundungsfehlern Schwierigkeiten bereiten [8]. Die Ursachen hierfür liegen vor allem in der möglicherweise unterschiedlichen Größenordnung der in der Regressionsrechnung auftretenden Zahlen und in der schlechten Konditionierung der zu invertierenden Präzisionsmatrix $\mathbf{X}^T \cdot \mathbf{X}$. Im folgenden werden kurz zwei Lösungsstrategien vorgestellt.
In MATLAB kann zur Schätzung von Parametern in linearen Gleichungssystemen der *backslash*-Operator „\“ eingesetzt werden. Dabei erhält man folgendermaßen die Parameterschätzwerte

$$\mathbf{b} = \mathbf{X} \setminus \mathbf{y} \tag{6-4}$$

Der *backslash*-Operator löst auf der Grundlage der Methode der kleinsten Fehlerquadrate lineare Gleichungssysteme, wobei verschiedene Algorithmen je nach Beschaffenheit des Systems eingesetzt werden [15].
Bei dem in [8] beschriebenen, alternativen Vorgehen werden zunächst die Werte der unabhängigen Variablen zentriert, d. h. man betrachtet nicht die Variablen selbst, sondern deren Abweichung vom entsprechenden Mittelwert. Die so erhaltene Regressionsgleichung kann um einen Parameter reduziert werden und wird mit den Standardabweichungen normiert. Der Koeffizientenvektor dieser transformierten Gleichung ergibt sich dann als Produkt aus der inversen Korrelationsmatrix und dem Vektor der Korrelationskoeffizienten zwischen den Antworten und den unabhängigen Variablen. Durch die Rücktransformation erhält man die Schätzwerte für die gesuchten Parameter.
Dieses Verfahren ist bei der Programmerstellung mit dem *backslash*-Operator von MATLAB verglichen worden, wobei sich der MATLAB-Operator als leistungsfähiger erwiesen hat und folglich auf eine Implementierung des Verfahrens nach [8] in der Programmsammlung verzichtet worden ist.

Für die Durchführung einer linearen Regression werden zwei Routinen zur Verfügung gestellt. Dabei handelt es sich um die Funktion *linreg* zur Parameterschätzung in Geradengleichungen und um *mlinreg* zur Durchführung einer multilinearen Regression. Beide Funktion führen die Parameterschätzung auf der Grundlage von Gl. (6-4) durch. Eine kurze Beschreibung dieser Routinen befindet sich in Tab. 6-1.

Tab. 6-1. Eigenschaften der Funktionen *linreg* und *mlinreg*

linreg:	Ermittlung von Geradenparametern (einfache lineare Regression)
Eingabe:	Matrix der unabhängigen Variablen, Antwortvektor, Konfidenzzahl*, graphische Darstellung (j/n)*
Ausgabe:	Steigung, Achsenabschnitt, Konfidenzintervallhalblänge der Steigung, Konfidenzintervallhalblänge des Achsenabschnitts, Konfidenzintervallhalblängen der Antworten, Korrelationskoeffizient der Stichprobe
mlinreg:	Durchführung einer multilinearen Regression
Eingabe:	Matrix der unabhängigen Variablen, Antwortvektor, Konfidenzzahl*
Ausgabe:	Parametervektor, Standardabweichung der Schätzung Konfidenzintervallhalblängen der Schätzung, Schätzung der Varianz, Standardabweichung der Antworten, Konfidenzintervallhalblängen der Antworten, Varianz-Kovarianz-Matrix der Schätzung, Residuen, Modellwerte der Antworten

* = Eingabe freigestellt

Die Funktionen *linreg* und *mlinreg* ermöglichen die Bestimmung verschiedener individueller Konfidenzgrößen, deren Berechnungsgrundlagen nach [8,10,32] hier vorgestellt werden sollen.
Die Schätzung der Varianz erfolgt in beiden Programmen über die Beziehung

$$\sigma^2 \approx s_U^2 = \frac{1}{n-m-1} \cdot \sum_{i=1}^{n} \left(y_i - \hat{y}_i\right)^2 \tag{6-5}$$

Mit Gl. (6-5) wird dann die Varianz-Kovarianz-Matrix der Schätzung ermittelt nach

$$\mathbf{V}_b = \left(\mathbf{X}^T \cdot \mathbf{X}\right)^{-1} \cdot s_U^2 \tag{6-6}$$

Die Diagonalelemente dieser Matrix entsprechen den Quadraten der Standardabweichungen der Schätzparameter s_b. Aus ihnen werden in Abhängigkeit der Freiheitsgrade und einer vorgegebenen Konfidenzzahl die *t*-verteilten Konfidenzintervallhalblängen l_b bestimmt nach

$$\left(l_b\right)_k = \left(s_b\right)_k \cdot t_\gamma\,(n-m-1) \tag{6-7}$$

Der Index *k* bezieht sich hierbei auf den *k*-ten Parameter.
Durch diese individuellen Vertrauensbereiche wird das Intervall festgelegt, in dem sich die wahren Werte mit einer vorgegebenen Wahrscheinlichkeit befinden. Analog dazu werden die Halblängen der Vertrauensintervalle l_y an einem

Versuchspunkt *i* mit dem Vektor der Einstellvariablen $\mathbf{x}_i$ für die Erwartungswerte der abhängigen Variablen ermittelt gemäß

$$\left(l_y\right)_i = \left(s_y\right)_i \cdot t_\gamma (n-m-1) = \sqrt{\mathbf{x}_i{}^T \cdot \left(\mathbf{X}^T \cdot \mathbf{X}\right)^{-1} \cdot \mathbf{x}_i} \cdot s_U \cdot t_\gamma (n-m-1) \qquad (6\text{-}8)$$

Über *linreg* kann im Gegensatz zu *mlinreg* zusätzlich die graphische Darstellung der Regressionsgeraden einschließlich der Auftragung der Vertrauensbereiche für die Erwartungswerte der abhängigen Variablen erzeugt werden. Für eine lineare Regression zur Bestimmung von Arrhenius-Parametern gemäß Kap. 7.7 ist dies beispielhaft in Abb. 6-1 dargestellt.

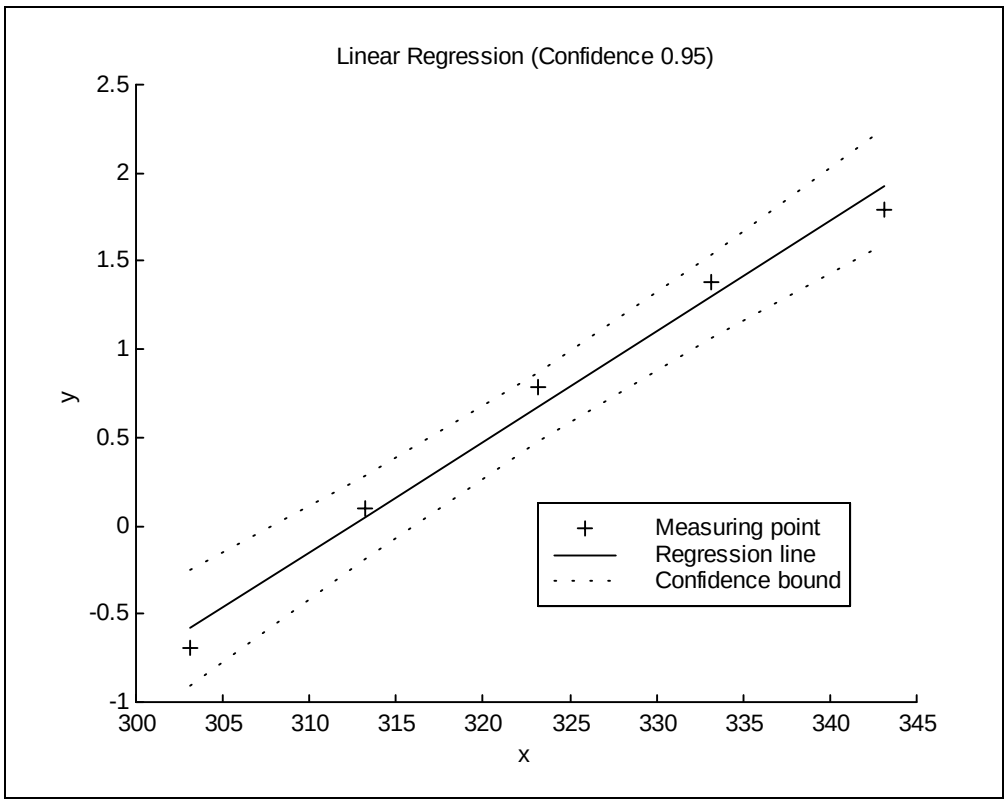

Abb. 6-1. Darstellung einer mit *linreg* in MATLAB ermittelten Regressionsgeraden und ihrer Vertrauensgrenzen

Das Ziel bei der Erstellung der vorliegenden Funktionen zur linearen Regression ist vorrangig gewesen, mit den Möglichkeiten von MATLAB Programmierabläufe zu vereinfachen, um auf diese Weise die wichtigsten Größen einer linearen Regression einfach und schnell zugänglich zu machen. Dadurch wird letztendlich auch die Vollständigkeit und Einheitlichkeit dieser Programmsammlung gewährleistet. Bezüglich weiterer Methoden zur linearen Regressionen sei auf die *Statistics Toolbox* und die *Chemometrics Toolbox* verwiesen [17,33].

6.2 Gemeinsame Vertrauensbereiche

Verzeichnis: *linear*
Funktionen: *lincorrp.m*
mlcorrp.m
mltestp.m

Die bisher behandelten individuellen Vertrauensbereiche sind direkt in den Programmen zur Parameterschätzung berechnet worden (vgl. Kap. 6.1). Die gemeinsamen Vertrauensbereiche können bei der Parameterschätzung in Geradengleichungen über die Funktion *lincorrp* und für die multilineare Regression durch die Funktion *mlcorrp* ermittelt werden (vgl. Kap. 2.4).
Die Eigenschaften dieser Funktionen sind in Tab. 6-2. zusammengefaßt.

Tab. 6-2. Eigenschaften der Funktionen zur Ermittlung und Überprüfung von gemeinsamen Vertrauensbereichen bei der linearen Regression

lincorrp:	Ermittlung und Darstellung gemeinsamer Vertrauensbereiche bei einer einfachen linearen Regression
Eingabe:	Matrix der unabhängigen Variablen, Antwortvektor, Schätzwert der Parameter, Konfidenzzahl*
Ausgabe:	Korrelationskoeffizient der Stichprobe, Schätzung der Fehlervarianz, Varianz-Kovarianz-Matrix der Schätzung, Standardabweichung der Schätzparameter, graphische Darstellung verschiedener Vertrauensintervalle
mlcorrp:	Ermittlung und Darstellung gemeinsamer Vertrauensbereiche bei einer linearen Regressionen
Eingabe:	Matrix der unabhängigen Variablen, Antwortvektor, Schätzwert der Parameter, Konfidenzzahl*, graph. darzustellende Parameter, Schnittniveau*
Ausgabe:	Schätzung der Fehlervarianz, Standardabweichung der Schätzparameter, Längen der Halbachsen des Vertrauensbereichshyperellipsoids, graphische Darstellung eines Niveauschnitts durch das Hyperellipsoid
mltestp:	Überprüfung, ob vorgegebene Parameter im gemeinsamen Vertrauensbereich liegen.
Eingabe	Matrix der unabhängigen Variablen, Antwortvektor, Schätzwert der Parameter, zu untersuchender Parametersatz, Konfidenzzahl*
Ausgabe	Schätzung der Fehlervarianz, Überprüfung, ob Parametersatz im gemeinsamen Vertrauensbereich liegt

* = Eingabe freigestellt

Die Grundlage für die Bestimmung gemeinsamer Vertrauensbereiche bildet die Quadratsummenzerlegung bzw. die Varianzanalyse. Dabei geht man davon aus,

daß sich die Abweichung vom wahren Modell durch die Abweichung von der Regression und durch die Abweichung aufgrund von Parameterungenauigkeiten ausdrücken läßt [8]. Als Ergebnis weitergehender Überlegungen ergibt sich die Begrenzungslinie des gemeinsamen Vertrauensbereichs der Modellparameter für ein vorgegebenes Vertrauensniveau γ näherungsweise als Höhenlinie der Residuenquadratsumme *SSQ*. Diese Höhenlinie stellt nach [8,34] eine *F*-verteilte Funktion der zur Schätzung herangezogenen Residuenquadrate dar gemäß

$$SSQ_\gamma = SSQ \cdot \left(1 + \frac{m}{n-m} \cdot F_\gamma (m, n-m)\right) \tag{6-9}$$

Die durch Gl. (6-9) beschriebenen Vertrauensbereiche entsprechen im linearen Fall in ihrer geometrischen Bedeutung den sogenannten Vertrauensbereichshyperellipsoiden und genügen folgender Beziehung [8,10]

$$(\mathbf{b} - \beta)^T \cdot \mathbf{X}^T \cdot \mathbf{X} \cdot (\mathbf{b} - \beta) = s_U^2 \cdot (m+1) \cdot F_\gamma (m, n-m-1) \tag{6-10}$$

Der Vorteil dieser Betrachtungen über die gemeinsamen Vertrauensbereiche besteht darin, daß die wahre Fehlervarianz σ^2 nicht bekannt sein muß.

Die Funktionen *mlcorrp* und *lincorrp* verfügen zwar über die gleichen theoretischen Grundlagen, aber unterscheiden sich sowohl in ihren Möglichkeiten als auch in ihrer programmtechnischen Umsetzung.
Im Gegensatz zu *mlcorrp* erfolgt mit *lincorrp* zusätzlich die Berechnung des Korrelationskoeffizienten und die Abfrage der Varianz-Kovarianz-Matrix der Schätzung (vgl. Tab. 6-2).
Die Ermittlung der Begrenzungslinie des Vertrauensbereichsellipsoids wird in *lincorrp* über den *matlab*-eigenen Höhenplotbefehl *contour* durchgeführt. Dazu wird in Abhängigkeit der individuellen Vertrauensintervalle ein Gitter über den Parameterraum gelegt und an den Gitterpunkten die Werte der linken Seite von Gl. (6-10) errechnet. Durch *contour* wird anschließend gemäß einer vorgegebenen Konfidenzzahl die Niveaulinie bestimmt, die Gl. (6-10) erfüllt.
Außerdem werden in der durch *lincorrp* erhaltenen graphischen Darstellung die individuellen Vertrauensintervalle den gemeinsamen gegenübergestellt.
Aufgrund der speziellen Abfassung der Funktion *lincorrp* ist ferner der Eingabeaufwand geringer als bei *mlcorrp* (vgl. Tab. 6-2).
Zur Demonstration werden in Abb. 6-2 die mit *lincorrp* ermittelten Vertrauensbreiche bezogen auf das Beispiel zu Abb. 6-1 dargestellt.

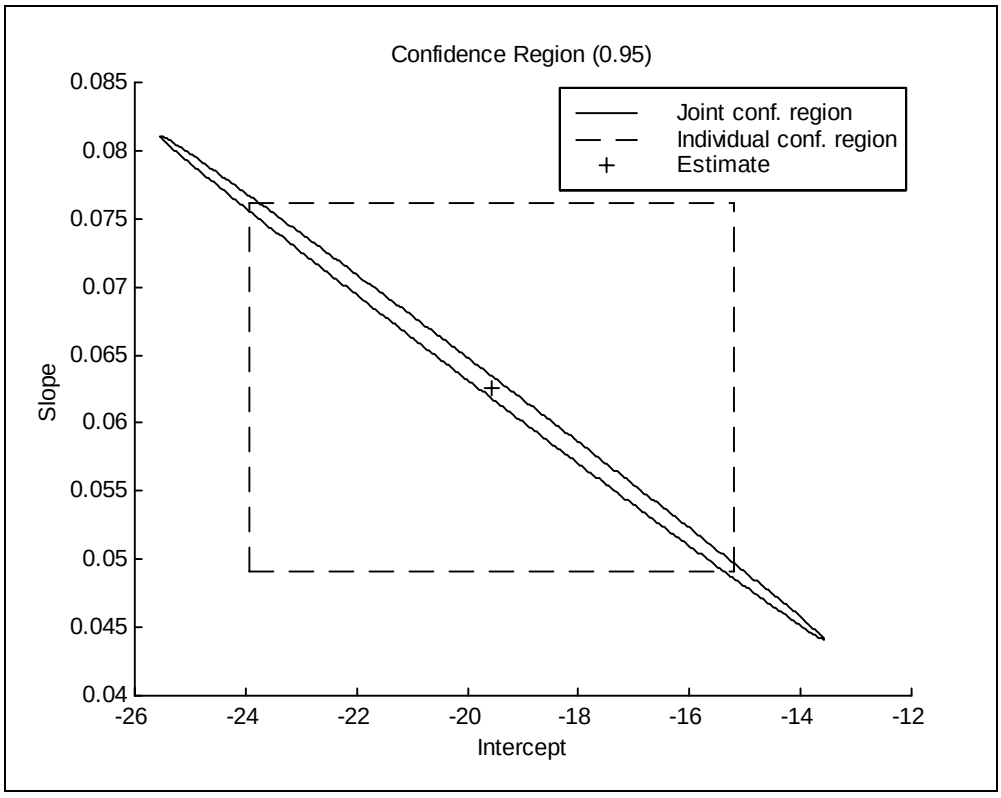

Abb. 6-2. Darstellung verschiedener mit *lincorrp* ermittelter Vertrauensbereiche zu einer linearen Regression in MATLAB (vgl. Abb. 6-1)

Bei *mlcorrp* ist die programmtechnische Umsetzung ähnlich wie bei *lincorrp*. Hier werden der Schätzwert der Varianz der Antworten und die Vertrauensintervallhalblängen berechnet. Ferner werden für das Hyperellipsoid die Halbachsenlängen bestimmt, indem die quadratische Form der Gl. (6-10) auf die kanonische Form reduziert wird [7,8].

Die graphische Darstellung des gemeinsamen Vertrauensbereichs erfolgt für jeweils zwei ausgewählte Parameter als Schnitt durch das Hyperellipsoid. Dabei können die übrigen Parameter extern vorgegeben werden, so daß das Niveau des Schnitts von außen festgelegt werden kann.

Für die Anwendung von *contour* wird programmintern eine der Dimension des Parametervektors angepaßte allgemeine und auf den Halbachsenraum transformierte Ellipsengleichung aufgestellt. Für diese wird der Niveauschnitt durchgeführt. Die graphischen Darstellungen erfolgen im Halbachsenraum, im Parameterraum und in dem auf den Ursprung transformierten Parameterraum. Die letzten beiden Darstellungen ermöglichen Betrachtungen zur räumlichen Lage des Hyperellipsoids. Sie ergeben sich aus der Transformation der Niveaulinie des Halbachsenraums in diese Räume. Auf die drei verschiedenen graphischen Darstellungen wird zurückgriffen, da sich bei der Programm-

erstellung gezeigt hat, daß MATLAB die Niveaulinien sehr viel besser um den Ursprung erstellen kann.

Mit der in Tab. 6-2 aufgeführten Funktion *mltestp* kann auf der Grundlage von Gl. (6-10) bestimmt werden, ob ein beliebiger Satz an Parametern in dem gemeinsamen Vertrauensbereich der geschätzten Parameter liegt.

6.3 Aspekte der Korrelationsanalyse

Verzeichnis: *variance*
Funktionen: *mcorrcoeff.m*
corrmat.m

Vorrangiges Ziel der Korrelationsanalyse ist die Aufklärung gegenseitiger Abhängigkeiten zwischen Zufallsvariablen, während die Regressionsanalyse sich weitgehend auf die Untersuchung einseitiger Abhängigkeiten beschränkt. Ein Maß für den Grad der gegenseitigen linearen Abhängigkeit ist der sogenannte Korrelationskoeffizient, der auch als Gütekriterium für eine durchgeführte Regression verwendet werden kann [8,10,32].
Da bei der Korrelationsanalyse die unabhängigen Variablen als Zufallsvariablen betrachtet werden, muß bei Annahme normalverteilter Variablen eine mehrdimensionale Normalverteilung betrachtet werden. Ein Parameter dieser Verteilung ist der Korrelationskoeffizient. Der Wert des Korrelationskoeffizienten kann jedoch nur aus den Meßwerten, der Stichprobe, geschätzt werden und ergibt sich als Stichprobenkorrelationskoeffizient r_{xy} für eindimensionale Modelle zu

$$r_{xy}^{\,2} = \frac{(\mathbf{y}-\bar{\mathbf{y}})^T \cdot (\mathbf{x}-\bar{\mathbf{x}}) \cdot (\mathbf{y}-\bar{\mathbf{y}})^T \cdot (\mathbf{x}-\bar{\mathbf{x}})}{(\mathbf{x}-\bar{\mathbf{x}})^T \cdot (\mathbf{x}-\bar{\mathbf{x}}) \cdot (\mathbf{y}-\bar{\mathbf{y}})^T \cdot (\mathbf{y}-\bar{\mathbf{y}})} \tag{6-11}$$

$\bar{\mathbf{x}}$ und $\bar{\mathbf{y}}$ sind hierbei die Mittelwerte der jeweiligen Stichprobe. Der Korrelationskoeffizient liegt zwischen +1 und –1, wobei die Korrelation um so stärker ist, je näher der Koeffizient bei +1 oder –1 liegt.

Bei Regressionsverfahren stellen die unabhängigen Variablen keine Zufallsvariablen dar, so daß der Korrelationskoeffizient aus den beobachteten Werten und den Modellwerten formuliert wird bzw. über deren Zusammenhänge bezüglich der unabhängigen Variablen aus Gl. (6-11) abgeleitet werden kann. Es gilt

$$R^2 = r_{y\hat{y}}{}^2 = \frac{(\hat{\mathbf{y}} - \overline{\mathbf{y}})^T \cdot (\hat{\mathbf{y}} - \overline{\mathbf{y}})}{(\mathbf{y} - \overline{\mathbf{y}})^T \cdot (\mathbf{y} - \overline{\mathbf{y}})} \tag{6-12}$$

R^2 stellt das Verhältnis aus erklärter und gesamter Variation dar. Dieser Korrelationskoeffizient R wird auch als multipler Korrelationskoeffizient bezeichnet, da er auf mehrdimensionale Modelle anwendbar ist. Falls die Regressionsgleichung die Meßwerte exakt wiedergibt, nimmt R^2 den Wert eins an.

MATLAB stellt in der Grundversion für diese Größen mit den Funktionen *var*, *cov* und *corrcoef* geeignetes Werkzeug zur Verfügung, das mit den Funktionen *mcorrcoeff* zur Berechnung des multiplen Korrelationskoeffizienten nach Gl. (6-12) und *corrmat* zur Aufstellung der Korrelationsmatrix im Rahmen dieser Arbeit ergänzt worden ist (vgl. Tab. 6-3).

Tab. 6-3. Eigenschaften der Funktionen *mcorrcoeff* und *corrmat*.

mcorrcoeff:	Berechnung des multiplen Korrelationskoeffizienten bei Regressionen
Eingabe:	zwei Vektoren gleicher Länge
Ausgabe:	multipler Korrelationskoeffizient
corrmat:	Aufstellung der Korrelationsmatrix
Eingabe:	zwei Matrizen gleicher Zeilenzahl
Ausgabe:	Korrelationsmatrix

Mit Hilfe der Funktion *corrmat* können die Korrelationskoeffizienten zwischen den Spalten von zwei beliebigen Matrizen mit gleicher Zeilenzahl ermittelt werden. Dabei ergibt sich der (*i*,*j*)-te Koeffizient der Korrelationsmatrix aus der *i*-ten Spalte der ersten Matrix und der *j*-ten Spalte der zweiten Matrix. Bezüglich der Größe der einsetzbaren Matrizen unterscheidet sich *corrmat* von der MATLAB-Funktion *corrcoef*, da in dieser Funktion die Matrizen gleich dimensioniert sein müssen. Bei den Berechnungen in *corrmat* wird auf die Unterroutinen *variance* und *mcovariance* zurückgegriffen, die allerdings mit den MATLAB Funktionen *var* und *cov* vergleichbar sind.

6.4 Statistische Tests

Verzeichnis: *confidence*
Funktionen: *corrtest.m*
sigtest.m
regtest.m

Die in die Programmsammlung eingebundenen statistischen Testverfahren sollen die Möglichkeit zur Überprüfung geben, ob verwendete Regressionsmodelle auf der Basis der Meßwerte gerechtfertigt oder aussagekräftig sind. Die Testverfahren sind für die einfache lineare Regression entwickelt worden, aber können auch für andere Formen der Regression verwendet werden [8,10].
Die zu den Testverfahren gehörenden Funktionen sind vorab in Tab. 6-4 aufgeführt und werden nachfolgend näher erläutert.

Tab. 6-4. Eigenschaften der Funktionen *corrtest*, *regtest* und *sigtest*.

corrtest:	Überprüfung der Korrelationsannahme bei der einfachen linearen Regression
Eingabe:	Vektor der unabhängigen Variablen, Vektor der abhängigen Variablen, Signifikanzzahl
Ausgabe:	Aussage, ob eine Korrelation vorliegt.
regtest:	Test zur Modellschwäche einer linearen Regression
Eingabe:	Vektor der unabhängigen Variablen, Vektor der abhängigen Variablen, Modellantworten, Parameteranzahl, Signifikanzzahl
Ausgabe:	Aussage über Richtigkeit des angenommenen Modells
sigtest:	Prüfung auf Signifikanz der einfachen linearen Regression
Eingabe:	Vektor der unabhängigen Variablen, Vektor der abhängigen Variablen, hypothetischer Wert der Steigung, Signifikanzzahl
Ausgabe:	Aussage über die Abhängigkeit der Antworten von den unabhängigen Variablen

6.4.1 Test beim Korrelationskoeffizienten

Dieses Testverfahren kann bei allen Systemen eingesetzt werden, die in ihrer Regressionsgleichung die Struktur einer Geradengleichung gemäß Gl. (6-13) aufweisen oder sich darauf transformieren lassen.

$$\hat{\mathbf{y}} = b_1 \cdot \mathbf{x} + b_0 \tag{6-13}$$

Der Grundgedanke dieses Signifikanztests ist die Annahme zu hinterfragen, ob die unabhängigen und abhängigen Variablen in der Form von Gl. (6-13) korreliert sind. Dazu wird der wahre Korrelationskoeffizient ρ statistisch betrachtet. Der dafür notwendige Algorithmus ist in Tab. 6-5. wiedergegeben. Er ist nur unter der Voraussetzung normalverteilter Variablen gültig.

Tab. 6-5. Überprüfung der Korrelationsannahme bei der einfachen linearen Regression nach [10]

Hypothese: Der Korrelationskoeffizient ρ sei 0 !
1. Festlegung einer Signifikanzzahl α (5%, 1% o. ä.)
2. Bestimmung des Arguments c der t-Verteilung für n-2 Freiheitsgrade bei einer Wahrscheinlichkeit von 1-α
3. Berechnung des Stichprobenkorrelationskoeffizienten r nach Gl. (6-11)
4. Berechnung von $t_0 = r \cdot \sqrt{\frac{n-2}{1-r^2}}$
5. Für $t_0 \le c$ wird die Hypothese angenommen.

Nach Vorgabe der Daten für die abhängigen und unabhängigen Variablen und der Signifikanzzahl wird der Algorithmus in der Funktion *corrtest* durchgeführt. Das Argument der t-Verteilung wird dabei über *tdistinv* (vgl. Kap 4.3) und der Korrelationskoeffizient mit der MATLAB-Funktion *corrcoef* ermittelt.

6.4.2 Test beim Regressionskoeffizienten

In diesem Testverfahren werden analog zum vorhergehenden Abschnitt ebenfalls nur Modelle gemäß Gl. (6-13) betrachtet. Hierbei soll die Signifikanz des eigentlichen Regressionskoeffizienten b_1 untersucht werden. Dafür ist die Routine *sigtest* entwickelt worden. Der zugrunde liegende Algorithmus ist in Tab. 6-6 aufgeführt und gilt nur, wenn die abhängigen Variablen für jede unabhängige Variable normalverteilt sind.

Für die Anwendung von *sigtest* müssen neben den Eingaben analog zu *corrtest* (x- und y-Werte, Signifikanzzahl) auch Angaben zum hypothetisch angenommenen Wert der Steigung gemacht werden (vgl. Tab. 6-4). Die unter Punkt 3. in Tab. 6-6 benötigten Größen werden durch die Funktionen *var* und *linreg* berechnet (vgl. Kap. 6.1 u. 6.3).
Dieses Verfahren wird bevorzugt eingesetzt, um zu überprüfen, ob die abhängigen y-Werte tatsächlich von den unabhängigen x-Werten abhängen. Dazu wird der Regressionskoeffizient b_1 mit Null angenommen.

Tab. 6-6. Prüfung auf Signifikanz der einfachen linearen Regression nach [10]

Hypothese: Der Regressionskoeffizient b_1 habe den Wert b_1^0!

1. Festlegung einer Signifikanzzahl α (5%, 1% oder o. ä.)
2. Bestimmung des Arguments c der t-Verteilung für n-2 Freiheitsgrade bei einer Wahrscheinlichkeit von 1-α
3. Berechnung der Standardabweichungen s_x und s_y der abhängigen und unabhängigen Variablen, sowie Steigung und Achsenabschnitt der Regressionsgeraden aus der Stichprobe.
4. Berechnung von

$$t_0 = s_x \cdot \sqrt{(n-1)\cdot(n-2)} \cdot \frac{b_1 - b_1^0}{\sqrt{b_0}}$$

5. Für $t_0 \le c$ wird die Hypothese angenommen.

6.4.3 Test zur Modellschwäche

Bisher ist die Richtigkeit der Modelle angenommen worden, die dem Regressionsverfahren unterzogen worden sind. Mit dem im folgenden vorgestellten Verfahren kann diese Annahme überprüft werden, wobei keine weiterführende Aussage bezüglich besser geeigneter Modelle getroffen wird.
Voraussetzung ist auch hier die Normalverteilung der Variablen sowie das Vorhandensein von Wiederholungsmessungen an den einzelnen Meßstellen.
Der theoretische Hintergrund dieses Verfahrens ist die Varianzanalyse bzw. Quadratsummenzerlegung [10]. Dabei wird eine gesamte Variation, in diesem Fall die Fehlerquadratsumme, in Teilvariationen zerlegt, über die dann Rückschlüsse auf das Modell gezogen werden können. Die Quadratsummenzerlegung erfolgt hier in die Variation der Gruppenmittelwerte $\bar{y}_i$ von der Regression (Gl. (6-14)) und in die Variation innerhalb der Gruppen (Gl. (6-15)).

$$SSL = \sum_{i=1}^{q} n_i \cdot \left(\bar{y}_i - \hat{y}_i\right)^2 \qquad (6\text{-}14)$$

$$SSG = \sum_{i=1}^{q} \sum_{j=1}^{n_i} \left(y_{ij} - \bar{y}_i\right)^2 \qquad (6\text{-}15)$$

Die Gruppen beinhalten jeweils die y-Werte, die zu einer Meßstelle x_i gehören. Die Gruppenmittelwerte stellen somit die Mittelwerte der y-Werte in einer

solchen Gruppe dar. Die Anzahl der Gruppen sei dabei q, die Anzahl ihrer Elemente n_i. Diese Gruppeneinteilung macht deshalb eine Doppelindizierung der y-Werte in Gl. (6-15) notwendig.
Die weiteren Untersuchungen zur Modellschwäche erfolgen durch den in Tab. 6-7 dargelegten Algorithmus. Dabei entspricht m der Anzahl der Regressionsparameter und n der Gesamtzahl an Versuchen.

Tab. 6-7. Test zur Modellschwäche einer linearen Regression nach [10]

Hypothese: Das Regressionsmodell ist korrekt!

1. Berechnung von SSL und SSG aus der Stichprobe und anschließend

$$v_0 = \frac{SSL \cdot (n-q)}{SSG \cdot (q-m-1)}$$

2. Festlegung einer Signifikanzzahl α (5%, 1% o. ä.)
3. Bestimmung des Arguments c der F-Verteilung für (q-m-1,n-q) Freiheitsgrade bei einer Wahrscheinlichkeit von 1-α
4. Für $v_0 \le c$ wird die Hypothese angenommen.

Die Umsetzung dieses Algorithmus erfolgt in der Funktion *regtest*, deren wichtigste Eigenschaften in Tab. 6-4 aufgeführt sind.
Für die Anwendung von *regtest* müssen der Vektor der unabhängigen Variablen und die dazugehörenden Antwortvektoren vorgegeben werden. Intern werden diese Vektoren mit der MATLAB-Routine *sortrows* sortiert und durch die eigens dafür entwickelte Funktion *numonum* aus dem Verzeichnis *general* in die oben erwähnten Gruppen eingeteilt. Anschließend werden die für Gl. (6-14) und Gl. (6-15) notwendigen Größen bestimmt, die entsprechenden Variationen berechnet und das Testverfahren durchgeführt.
Die Funktion *regtest* kann prinzipiell auch auf bezüglich der unabhängigen Variablen mehrdimensionale Modelle angewendet werden. Dazu werden lediglich die Werte einer unabhängigen Variablen als Vektor vorgegeben, um die angeführten Sortierungen für die Berechnungen zu ermöglichen.

7 Nichtlineare Regression bei Einfachantwortsystemen

In diesem Abschnitt werden nichtlineare Regressionen bei Einfachantwortsystemen behandelt (vgl. Kap. 6). Diese Form der Regressionsrechnung hat für die Reaktionskinetik eine besondere Bedeutung, da viele mechanistische Modelle nichtlinear bezüglich ihrer Parameter sind [9,25].

Die Systeme, die in diesem Kapitel betrachtet werden, beschränken sich damit analog zu Gl. (2-1) auf folgende Struktur:

$$\eta = f(\boldsymbol{\beta}, \mathbf{x}) \tag{7-1}$$

Entsprechend gilt für die Regressionsgleichung nach Gl. (2-3)

$$\hat{y} = f(\mathbf{b}, \mathbf{x}) \tag{7-2}$$

Die Schwierigkeit der Parameterschätzung in diesen Modellen besteht darin, daß die Konvergenz der Optimierverfahren wegen der Nichtlinearität gefährdet und die Antwortfläche durch eine Vielzahl an lokalen Optima geprägt sein kann.
Für die Durchführung derartiger Regressionen werden in der Programmsammlung zwei Lösungsstrategien zur Verfügung gestellt. Zum einen erfolgt die Parameterbestimmung über eine Linearisierung der Modellgleichung in den Parametern, zum anderen wird in einer allgemeineren Behandlung auf *matlab*-eigene Optimierverfahren zurückgegriffen.
Ferner wird in diesem Zusammenhang die Arrhenius-Gleichung wegen ihrer großen Bedeutung in der chemischen Kinetik als Spezialfall einer nichtlinearen Regression behandelt.
Grundsätzlich können die im folgenden näher erläuterten Methoden und Verfahren auch bezüglich einer Parameterschätzung in linearen Modellen nach Gl. (6-1) eingesetzt werden. Allerdings können hierbei zusätzliche Fehlerquellen, wie z. B. Rundungsfehler, auftreten und sich die Rechenzeiten wegen der Verwendung anderer Optimierverfahren erhöhen.

7.1 Linearisierung der Modellgleichung in den Parametern

Verzeichnis: *linearization*
Funktionen: *lininp.m*
lpgaunew.m

Für die Linearisierung einer Modellgleichung in den Parametern mit anschließender Parameterschätzung wird die Funktion *lininp* zur Verfügung gestellt. Die Grundlage dieser Funktion ist ein vereinfachtes Gauß-Newton-Verfahren. Dazu wird die Modellgleichung in eine Taylor-Reihe überführt, die nach dem linearen Term abgebrochen wird. Allgemein läßt sich damit die linearisierte Modellgleichung mit p Parametern an einem Versuchspunkt i nach [9,20] formulieren als

$$\tilde{\eta}_i = \hat{y}_i + \sum_{k=1}^{p} \frac{\partial \eta_i}{\partial \beta_k} \cdot (\beta_k - b_k) = \hat{y}_i + \mathbf{J}_i \cdot (\beta - \mathbf{b}) \qquad (7\text{-}3)$$

Die Parameterwerte b_k sind die Schätzwerte für die Linearisierungsstelle.
Setzt man für $\tilde{\eta}_i$ in Gl. (7-3) die Beobachtungswerte ein, so läßt sich die Beziehung auf die Form von Gl. (6-1) bringen und ist damit mit den bereits vorgestellten Methoden der linearen Regression lösbar (vgl. Kap. 6).
Die Linearisierung wird in *lininp* unter Verwendung der Funktionen *jacsym* und *jacsymval* zur Berechnung der Jacobi-Matrix durchgeführt. Die Parameterschätzung erfolgt über den *backslash*-Operator (vgl. Kap. 4.2 u. 6.1). Die Eigenschaften von *lininp* sind in Tab. 7-1 aufgelistet.

Tab. 7-1. Eigenschaften der Funktionen *lininp* und *lpgaunew*

lininp:	Linearisierung nichtlinearer Modelle mit anschl. Parameterschätzung
Eingabe:	Matrix der unabhängigen Variablen, Antwortvektor, Modellgleichung, Linearisierungsstelle*
Ausgabe:	Parameterschätzwerte, Schätzwert für die Fehlervarianz, analytische Form der Jacobi-Matrix (*cell array*), Modellantworten, Residuen, Jacobi-Matrix und Modellwerte an der Linearisierungsstelle
lpgaunew:	Parameterschätzung in nichtlinearen Modellen auf der Basis des Gauß-Newton-Verfahrens
Eingabe:	siehe *lininp*, zusätzl. Anzahl der Iterationsschritte
Ausgabe:	siehe *lininp*, zusätzl. Parameterschätzwerte, Fehlerquadratsummen zu jedem Iterationsschritt

* = Eingabe freigestellt

Die Anwendung der Funktion *lininp* entspricht geometrisch im Parameterraum der Überführung einer gekrümmten Modellfläche in eine Tangentialebene. Die Parameterschätzung ist dabei um so genauer, je mehr der Berührpunkt beider Flächen mit dem wahren Wert des Parametervektors übereinstimmt. Da die Vorgabe der Linearisierungsstelle in *lininp* auf möglicherweise sehr ungenauen Annahmen beruht, muß beim Gauß-Newton-Verfahren iteriert werden, um die

Parameterschätzung schrittweise zu verbessern. Dieses geschieht hier mittels der Funktion *lpgaunew* (vgl. Tab. 7-1).
In *lpgaunew* wird die Anzahl der durchzuführenden Iterationsschritte vorgegeben und die Routine *lininp* in einer Schleife aufgerufen. Der in einem Schleifenaufruf berechnete Schätzwert wird dann modifiziert und ist im nächsten Schritt die neue Linearisierungsstelle für *lininp*. Die Modifizierung des Schätzvektors erfolgt, indem an dem eigentlichen Schätzvektor **b** die Gauß-Newton-Korrektur durchgeführt wird. Somit gilt nach [9] für den i-ten und (i+1)-ten Durchlauf

$$\mathbf{b}^{(i+1)} = \mathbf{b}^{(i)} + \Delta\mathbf{b}^{(i)} \tag{7-4}$$

mit der Gauß-Newton-Korrektur

$$\Delta\mathbf{b}^{(i)} = \left[\left(\mathbf{J}^T \cdot \mathbf{J}\right)^{-1} \mathbf{J}^T \cdot \left(\mathbf{y} - \hat{\mathbf{y}}\right)\right]^{(i)} \tag{7-5}$$

Das Gauß-Newton-Verfahren kann Konvergenzschwierigkeiten aufweisen [9]. Deshalb erfolgt der Abbruch bzw. die Beendigung des Verfahrens über die Anzahl der Iterationsschritte, um auf diese Weise gegebenenfalls einen externen Eingriff zum Abbruch der Rechnungen zu vermeiden. Um dennoch einen Einblick in die Konvergenz des Verfahrens zu haben, werden die Fehlerquadratsumme und die Schätzparameter in jedem Schritt abgespeichert und können bei Bedarf abgerufen werden.
Die Aufstellung der Jacobi-Matrix erfolgt in *lininp* und damit während eines jeden Iterationsschritts. Dieses erhöht zwar den Rechenaufwand, aber ermöglicht den separaten Einsatz von *lininp*, z. B. zur Aufstellung der linearisierten Form einer Modellgleichung.

Ein wichtiges Einsatzgebiet der Routinen *lininp* und *lpgaunew* besteht in der Schätzung akzeptabler Startwerte für die Durchführung einer nichtlinearen Regression (vgl. Kap 7.3 u. 12).
Auf die Implementierung weiterer Methoden zur iterativen Bestimmung von Regressionsparametern ist verzichtet worden, da diese Methoden, wie z. B. die Newton- oder Gradientenverfahren, in ihren Algorithmen nicht direkt auf die Modellgleichungen, sondern auf die Bearbeitung der Schätzfunktion zurückgreifen [35]. Derartige Verfahren sind teilweise in den MATLAB-Routinen umgesetzt und werden entsprechend in Kap. 7.3 behandelt.
MATLAB stellt in der *Statistics Toolbox* ebenfalls eine Schätzfunktion auf der Basis des Gauß-Newton-Verfahren zur Verfügung. Dabei werden die linearisier-

ten Gleichungen jedoch nicht in analytischer Form zur Verfügung gestellt und die Vertrauensbereiche nicht so detailliert ermittelt wie in Kap. 7.2 [17,36].

7.2 Vertrauensbereiche in linearisierten Modellen

Verzeichnis: *linearization*
Funktionen: *lpconfp.m*
lpcorrp.m
lptestp.m

Über die Funktion *lpconfp* erfolgt die Ermittlung der individuellen Vertrauensbereiche von Parametern in linearisierten nichtlinearen Modellen. Zu diesen Vertrauensintervallen zählen die Standardabweichungen der Parameter, die Konfidenzintervalle und Tangentialebenenvertrauensintervalle sowie die Varianz-Kovarianz-Matrix der Schätzung (vgl. Tab 7-2).

Tab. 7-2. Eigenschaften der Funktionen *lpconfp*, *lpcorrp* und *lptestp*

lpconfp:	Bestimmung individueller Vertrauensintervalle von Parametern in linearisierten nichtlinearen Modellen bei Einfachantwortsystemen (vgl. *lininp*)
Eingabe:	Vektor der Parameterschätzwerte, Jacobi-Matrix der Parameter für die Schätzwerte, Fehlervarianz, Konfidenzzahl*
Ausgabe:	Standardabweichungen, Konfidenzintervallhalblängen, Tangentialebenenvertrauensintervallhalblängen
lpcorrp:	Darstellung des gemeinsamen Vertrauensbereichs von zwei ausgewählten Parametern bei Schätzung in linearisierten nichtlinearen Modellen
Eingabe:	Vektor der Parameterschätzwerte, Jacobi-Matrix der Parameter für die Schätzwerte, Fehlervarianz, Standardabweichungen der Schätzparameter, Konfidenzzahl*, graphisch darzustellende Parameter, Schnittniveau*
Ausgabe:	Halblängen des Konfidenzellipsoids, Matrix des Niveauplots (vgl. MATLAB / *contour*), graph. Darstellung
lptestp:	Überprüfung, ob vorgegebene Parameter im gemeinsamen Vertrauensbereich liegen.
Eingabe:	Parameterschätzwerte, Jacobi-Matrix der Parameter für die Schätzwerte, Fehlervarianz, Parametersatz, Konfidenzzahl*
Ausgabe:	Schätzung der Fehlervarianz, Aussage, ob Parametersatz im gemeinsamen Vertrauensbereich liegt.

* = Eingabe freigestellt

Die Vorgehensweise in der Funktion *lpconfp* ist analog zu Kap. 6.1, allerdings wird statt Gl. (6-6) folgende Beziehung nach [9] für die Varianz-Kovarianz-Matrix der Schätzung $\mathbf{V}_b$ bei p Parametern und n Versuchen verwendet

$$\mathbf{V}_b = \left(\mathbf{J}^T \cdot \mathbf{J}\right)^{-1} \cdot s_U^2 = \left(\mathbf{J}^T \cdot \mathbf{J}\right)^{-1} \cdot \frac{1}{n-p} \cdot \sum_{i=1}^{n} \left(y_i - \hat{y}_i\right)^2 \tag{7-6}$$

Die Standardabweichungen der Parameter s_b und die Halblängen der Konfidenzintervalle werden nach Gl. (6-7) ermittelt und die Halblängen der Tangentialebenenvertrauensintervalle tl_b für den k-ten Parameter gemäß [9] nach

$$\left(tl_b\right)_k = \left(s_b\right)_k \cdot \sqrt{p \cdot F_\gamma\left(p, n-p\right)} \tag{7-7}$$

Die über die Tangentialebenen des Konfidenzellipsoids erhaltenen Vertrauensintervalle stellen das maximale individuelle symmetrische Vertrauensintervall dar, innerhalb dessen ein Parameter β_k mit einer bestimmten Wahrscheinlichkeit liegt.

Die Berechnung der gemeinsamen Vertrauensbereiche und ihrer graphischen Darstellung über die Routine *lpcorrp* ist vergleichbar zu der entsprechenden Funktion *mlcorrp* aus Kap. 6.2. Auch hier werden die Halbachsenlängen des Vertrauensbereichshyperellipsoids bestimmt und wird über den Höhenplotbefehl *contour* der Schnitt durch das Ellipsoid im Halbachsenraum, im Parameterraum und im transformierten Parameterraum für jeweils zwei vorgegebene Parameter dargestellt. Die Definitionsgleichung des Ellipsoids ist vergleichbar mit Gl. (6-10) und lautet

$$\left(\mathbf{b} - \boldsymbol{\beta}\right)^T \cdot \mathbf{J}^T \cdot \mathbf{J} \cdot \left(\mathbf{b} - \boldsymbol{\beta}\right) = s_U^2 \cdot p \cdot F_\gamma\left(p, n-p\right) \tag{7-8}$$

Ob ein Satz von Parametern im Vertrauensbereich der geschätzten Werte liegt, kann mit der Funktion *lptestp* überprüft werden (vgl. Tab. 7-2).
Bei den hier vorgestellten Verfahren zur Berechnung und Darstellung individueller und gemeinsamen Vertrauensintervalle ist zu beachten, daß sich bei besonders stark nichtlinearen Modellen die Vertrauensbereiche des linearisierten und nichtlinearen Modells stark unterscheiden können [9,34].

7.3 Allgemeine Behandlung nichtlinearer Regressionen

Verzeichnis: *singleresponse*
Funktionen: *singleres.m*

Für die allgemeine Behandlung nichtlinearer Regressionen bei Einfachantworten steht die Funktion *singleres* zur Verfügung. Diese Funktion unterscheidet sich bei der Parameterschätzung von den bisher vorgestellten Verfahren durch den Einsatz von numerischen Optimierverfahren und der Möglichkeit zur Wahl des Schätzkriteriums (vgl. Kap. 3.5).
Mit *singleres* erhält man die Paramerschätzwerte, eine Schätzung der Fehlervarianz, die dazugehörige Fehlerquadratsumme und die Residuen. Neben den üblichen Vorgaben können bei der Anwendung Angaben zur Varianz-Kovarianz-Matrix der Meßwerte und zu den Optimiereinstellungen berücksichtigt werden (vgl. Tab. 7-3).

Tab. 7-3. Eigenschaften der Funktion *singleres*

***singleres*:**	Parameterschätzung in nichtlinearen Modellen bei Einfachantwortsystemen
Eingabe:	Matrix der unabhängigen Variablen, Antwortvektor, Modellgleichung, Gewichtsvektor*, Vektor der Startschätzwerte, Optimieroptionen*, Optimierverfahren*, Schätzfunktion*, Beschränkung des Parameterraums (nur *constr*)*, Nebenbedingung (nur *constr*)*
Ausgabe:	Parameterschätzwerte, Schätzung der Fehlervarianz, Wert der Schätzfunktion, Residuen

* = Eingabe freigestellt

In der programmtechnischen Umsetzung greift die Hauptfunktion *singleres* auf drei Unterprogramme zurück. Zunächst wird für den Aufruf des Optimierverfahrens die Zentralfunktion *optrout* aufgerufen. Das dadurch aktivierte Optimierverfahren greift auf die Zielfunktion *sroptfcn* zurück. In der Unterfunktion *sroptfcn* wird über die Zentralfunktion der Schätzung *srobjfcn* der Wert des Schätzkriteriums an das Optimierverfahren übermittelt (vgl. Kap. 3.5). Die Programmstruktur gliedert sich damit analog zu Abb. 3-2.
Als Optimierverfahren können standardmäßig die MATLAB-Verfahren *fmins*, *leastq* und *constr* herangezogen werden. Bei Verwendung von *constr* können bei Einsatz von *singleres* weitergehende Angaben bezüglich der Beschränkung des Parameterraums und von Nebenbedingungen gemacht werden. (vgl. Kap 3.5).

Die über die Zentralfunktion *srobjfcn* nach [11] voreingestellten Schätzkriterien sind in Tab. 7-4 aufgeführt.

Tab. 7-4. Schätzfunktionen für die nichtlineare Regression bei Einfachantwortsystemen

Bezeichnung	Beschreibung	Schätzfunktion Ψ
Φ_1 ('1')	Normalverteilung ungewichtete Fehlerquadratsumme	$\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$
Φ_2 ('2')	Normalverteilung, gewichtete Fehlerquadratsumme (Gewichtsvektor **w**, extern vorgegeben)	$\sum_{i=1}^{n} w_i \cdot (y_i - \hat{y}_i)^2$

Unter den Schätzfunktionen ermöglicht das Kriterium Φ_2 im Gegensatz zu Φ_1 eine Gewichtung der Meßwerte. Die Schätzfunktionen können wie in Kap. 3.5.2 beschrieben ergänzt werden.

7.4 Vertrauensbereiche bei nichtlinearen Modellen

Verzeichnis: *singleresponse*
Funktionen: *srconfp.m*
srcorrp.m

Die Berechnung und graphische Darstellung von Vertrauensintervallen kann für die individuellen Vertrauensintervalle mit der Funktion *srconfp* und für die gemeinsamen Vertrauensbereiche mit *srcorrp* erfolgen (vgl. Kap. 6 u. 7.2). Einzelheiten zu diesen Funktionen sind Tab. 7-5 dargelegt.

Tab. 7-5. Eigenschaften der Funktionen *srconfp* und *srcorrp*

srconfp:	Bestimmung individueller linearisierter Vertrauensintervalle von Parametern in nichtlinearen Modellen bei Einfachantwortsystemen
Eingabe:	Matrix der unabhängigen Variablen, Antwortvektor, Schätzvektor der Parameter, Modellgleichung, gewählte Schätzfunktion*, Gewichtsvektor*, Konfidenzzahl*
Ausgabe:	Parameterschätzwerte, Varianz-Kovarianz-Matrix der Schätzung, Standardabweichung, Konfidenzintervallhalblängen, Halblängen des Tangentialebenenvertrauensintervalls (jeweils bezogen auf die Parameter), Wert der Schätzfunktion, Gradient, Jacobi-Matrix, Hesse-Matrix (jeweils bezogen auf die Parameter)

Forts. Tab.7-5.

srcorrp:	Darstellung des konturgetreuen gemeinsamen Vertrauensbereichs von zwei ausgewählten Parametern in nichtlinearen Modellen bei Einfachantwortsystemen
Eingabe:	Matrix der unabhängigen Variablen, Antwortvektor, Schätzvektor der Parameter, Modellgleichung, Fehlervarianz, graphisch darzustellende Parameter, zulässiger Parameterbereich*, Anzahl der Gitterpunkte je Achse*, Konfidenzzahl*, graph. Darstellung (j/n)*
Ausgabe:	Matrix der Niveaupunkte, Parameterwerte

* = Eingabe freigestellt

Mit der Funktion *srconfp* werden nicht nur individuelle Vertrauensintervalle bestimmt, sondern können auch wichtige Zwischenergebnisse, wie der Gradient der Schätzfunktion, die Jacobi-Matrix der Modellgleichungen und die Hesse-Matrix, jeweils auf die Parameter bezogen, abgerufen werden.
Um den Benutzer in der Freiheit bei der Ergänzung der Schätzfunktionen nicht einzuschränken, werden der Gradient, die Hesse-Matrix und die Varianz-Kovarianz-Matrix der Schätzung nach [7] allgemein berechnet.
Danach gilt für den Gradienten $\mathbf{q}$

$$\mathbf{q} = -2 \cdot \sum_{i=1}^{n} \mathbf{J}_i^T \cdot \boldsymbol{\Gamma} \cdot \left(y_i - \hat{y}_i \right) \tag{7-9}$$

für die Hesse-Matrix $\mathbf{H}$

$$\mathbf{H} \approx 2 \cdot \sum_{i=1}^{n} \mathbf{J}_i^T \cdot \boldsymbol{\Gamma} \cdot \mathbf{J}_i \tag{7-10}$$

und für die Varianz-Kovarianz-Matrix der Schätzung $\mathbf{V}_b$

$$\mathbf{V}_b \approx \left(\sum_{i=1}^{n} \mathbf{J}_i^T \cdot \boldsymbol{\Gamma} \cdot \mathbf{J}_i \right)^{-1} \cdot \left(\sum_{i=1}^{n} \mathbf{J}_i^T \cdot \boldsymbol{\Gamma} \cdot \mathbf{V} \cdot \boldsymbol{\Gamma} \cdot \mathbf{J}_i \right) \cdot \left(\sum_{i=1}^{n} \mathbf{J}_i^T \cdot \boldsymbol{\Gamma} \cdot \mathbf{J}_i \right)^{-1} \tag{7-11}$$

Dabei handelt es sich bei $\mathbf{J}_i$ um die Jacobi-Matrix am i-ten Versuchspunkt und bei $\mathbf{V}$ um die Varianz-Kovarianz-Matrix der Meßfehler, die bei Einfachantwortsystemen der Fehlervarianz entspricht. $\boldsymbol{\Gamma}$ stellt die Ableitung der Schätzfunktion nach der Fehlerquadratsumme dar. Um $\boldsymbol{\Gamma}$ zu berechnen, ist es notwendig die Schätzfunktion als Funktion der Fehlerquadratsumme zu formulieren (vgl. Kap. 8). Für Einfachantwortsysteme, die nach der Methode der kleinsten Fehlerquadrate nach Tab. 7-4 geschätzt werden, gilt

$$\boldsymbol{\Gamma} = 1 \tag{7-12}$$

Der funktionale Zusammenhang für $\boldsymbol{\Gamma}$ muß dabei in der Funktion *srobjfcn* festgelegt werden (vgl. Kap. 3.5.2).
Die Berechnung der individuellen Vertrauensintervalle erfolgt analog zu Kap. 7.2. Dieses bedeutet, die Standardabweichungen ergeben sich aus den Diagonalelementen der Varianz-Kovarianz-Matrix, während die *t*-verteilten Konfidenzintervallhalblängen und die *F*-verteilten Tangentialebenenvertrauensintervalle nach den Gl. (6-7) und Gl. (7-7) bestimmt werden.

Die Ermittlung und graphische Darstellung gemeinsamer Vertrauensbereiche erfolgt über die Funktion *srcorrp* (vgl. Tab. 7-5). Der Rechenablauf und die programmtechnische Umsetzung in MATLAB sind analog zu den Funktionen *lpcorrp* oder *mlcorrp* (vgl. Kap. 7.2 u. 6.2).
Bei den über *srcorrp* erhaltenen Vertrauensbereichen handelt es sich um konturgetreue gemeinsame Vertrauensbereiche, die durch Anwendung von Gl. (6-9) berechnet werden [9,34]. Dazu wird über den durch zwei ausgewählte Parameter festgelegten Parameterraum ein Gitter gelegt und an den Gitterpunkten die Fehlerquadratsumme bestimmt. Mittels der Funktion *contour* wird nun die Niveaulinie ermittelt, die Gl. (6-9) erfüllt. Dabei wird die graphische Darstellung nicht über den Höhenplotbefehl *contour* selbst vorgenommen, sondern die darüber erhaltene Matrix zwecks besserer Darstellung weiterverarbeitet.
Um die zur Berechnung der Höhenlinie notwendigen Modellantworten zu bestimmen, werden in der Modellgleichung für die übrigen Parameter die aus der Parameterschätzung erhaltenen Schätzwerte eingesetzt. Dieses bedeutet, Niveauschnitte wie mit der Funktion *mlcorrp* sind nicht möglich (vgl. Kap. 6.2).
Zur Erleichterung der Ermittlung des Konturenverlaufs können Zusatzinformationen beim Aufruf von *srcorrp* berücksichtigt werden. Dazu gehören die Festlegung des zulässigen Parameterbereichs oder die Anzahl der Gitterpunkte je Koordinatenachse. Die Bedeutung dieser Argumente ist bei Anwendung von *srcorrp* nicht zu unterschätzen, da ein zu grobmaschiges Gitter die Konturen des Vertrauensbereich nur ungenügend wiedergibt und ein zu klein vorgegebener Parameterbereich möglicherweise einen fälschlicherweise geschlossenen Kurvenverlauf anzeigt. Hinweise auf diese Fehlerquellen sind Kanten oder Nicht-Differenzierbarkeitsstellen im Kurvenverlauf. Beide Einstellparameter besitzen auch einen kombinierten Einfluß. Bei einer direkten Anwendung des *contour*-Befehls kann diese Problematik ebenfalls nicht ohne weiteres umgangen werden.

Die individuellen konturgetreuen Vertrauensintervalle können über die Matrix der Niveaupunkte bestimmt werden. Diese Matrix wird aus dem *contour*-Aufruf erhalten, über *srcorrp* ausgegeben und kann weiterverarbeitet werden. Für den Umgang mit dieser Matrix sei auf [37] verwiesen. Stellt man diese Niveaumatrix graphisch dar und transformiert das Koordinatensystem so, daß der Punkt der Parameterschätzwerte in den Ursprung fällt, stellen die Schnittpunkte mit den Koordinatenachsen die Größe der konturgetreuen Vertrauensintervalle dar.

Im Gegensatz zu *lpcorrp* werden in *srcorrp* keine Vereinfachungen zur Ermittlung der Niveaulinie gemacht, was sich vor allem in einer erhöhten Rechenzeit äußert (vgl. Kap. 7.2).

7.5 Anwendungsbeispiel

Zur Überprüfung der Einsatzfähigkeit der Programme ist ein Beispiel aus [9] entnommen worden. In diesem Beispiel wird eine Hydrierreaktion betrachtet, die durch folgendes Modell beschrieben wird

$$\eta = r = \frac{\beta_1 \cdot p}{1 + \beta_2 \cdot p} \tag{7-13}$$

Dabei steht p für den Partialdruck und β_1, β_2 für die Geschwindigkeitskonstanten. In der ybx-Schreibweise lautet Gl. (7-13)

$$y = \frac{b(1) \cdot x}{1 + b(2) \cdot x} \tag{7-14}$$

Die dazugehörigen Meßwerte sind in Tab.7-6 aufgeführt.

Tab. 7-6. Meßwerte für eine Hydrierreaktion (x Druck, y Reaktionsgeschwindigkeit)

x	y
20	0.0680
30	0.0858
35	0.0939
40	0.0999
50	0.1130
55	0.1162
60	0.1190

Die Parameterbestimmung erfolgte mit den Funktionen *lpgaunew* und *singleres*, d. h. zum einen durch Linearisierung der Modellgleichung in den Parametern mit anschließender linearer Regression, zum anderen durch eine nichtlineare Regression (vgl. Kap. 7.1 u. 7.3).
Die Startwerte für beide Verfahren sind folgendermaßen gesetzt worden:

$$b(1) = 0.001 \qquad b(2) = 0.01 \tag{7-15}$$

Bei der Anwendung von *lpgaunew* ist die Optimierung auf zehn Iterationsschritte begrenzt worden.
Die Optimierung über *singleres* ist mit *fmins* bei Standardeinstellungen und der ungewichteten Fehlerquadratsumme als Schätzfunktion durchgeführt worden.
Die individuellen linearisierten Vertrauensbereiche sind mit *lpconfp* bzw. *srconfp* berechnet und die Halbachsenlängen des jeweiligen Vertrauensbereichsellipsoids sind mit *lpcorrp* bestimmt worden. Die Ergebnisse sind mit den Literaturwerten zusammen in Tab. 7-7 aufgeführt.

Tab. 7-7. Ergebnisse der Parameterschätzung bezüglich des Geschwindigkeitsgesetzes einer Hydrierreaktion (**b** Parameterschätzwerte, *SSR* Summe der Residuenquadrate, s_U^2 Fehlervarianz, $\mathbf{s}_b$ Standardabweichungen der Parameter, $\mathbf{l}_b$ Halblängen der Konfidenzintervalle, $\mathbf{tl}_b$ Halblängen der Tangentialebenenvertrauensintervalle, $\mathbf{xl}_b$ Länge der Halbachsen des Konfidenzellipsoids)

	lpgaunew	*singleres*	Literatur [9]
b	5.1542*e*-3 2.6282*e*-2	5.1569*e*-3 2.6308*e*-2	5.154*e*-3 2.628*e*-2
SSR	4.7604*e*-4	4.7608*e*-6	4.758*e*-6
s_U^2	9.5209*e*-7	9.5217*e*-7	9.555*e*-7
$\mathbf{s}_b$	1.3486*e*-4 1.2736*e*-3	1.3499*e*-4 1.2748*e*-3	– –
$\mathbf{l}_b$ (95%)	3.4668*e*-4 3.2740*e*-3	3.4699*e*-4 3.2770*e*-3	3.466*e*-4 3.274*e*-3
$\mathbf{tl}_b$ (95%)	4.5878*e*-4 4.3326*e*-3	4.5919*e*-4 4.3367*e*-3	– –
$\mathbf{xl}_b$ (95%)	6.3762*e*-5 4.3564*e*-3	6.3797*e*-5 4.3604*e*-3	– –

Die Ergebnisse der beiden durchgeführten Rechnungen stimmen in guter Näherung mit den über ein Marquardt-Verfahren ermittelten Literaturwerten überein [9].
Auch der Vergleich der Ergebnisse innerhalb eines Verfahrens bestätigt die korrekte Umsetzung der theoretischen Grundlagen in den hier eingesetzten Programmen. So sind die Standardabweichungen als Konfidenzintervalle bei 68%-igen Vertrauen entsprechend kleiner als die Konfidenzintervalle bei 95%-igem Vertrauen, die wiederum von den Tangentialebenenvertrauensintervallen als den maximalen individuellen Vertrauensintervallen erwartungsgemäß eingeschlossen werden.

Vergleichbare Rechnungen sind auch mit anderen Startwerten durchgeführt worden. Bei Startparametern, die um den Faktor zehn von den oben angegebenen Startwerten abwichen, konnten bei den Rechnungen mit *lpgaunew* keine Ergebnisse mehr erhalten werden, da schon nach drei Iterationsschritten das Verfahren nicht mehr konvergierte. Dagegen sind mit *singleres* auch mit auf eins gesetzten Startwerten vergleichbare Ergebnisse erzielt worden, was für die Robustheit der in MATLAB implementierten Funktion *fmins* spricht.

Die gemeinsamen Vertrauensbereiche sind mit *lpcorrp* und *srcorrp* erstellt worden. Dabei ist der Rechenaufwand und damit die Rechenzeit wegen der Vereinfachungen bei *lpcorrp* geringer als bei *srcorrp* (vgl. Kap.7.2). Die individuellen Konfidenzintervalle sowie die elliptischen und konturgetreuen Vertrauensbereiche sind beispielhaft für die *singleres*-Rechnungen in Abb. 7-1 dargestellt. Für die Berechnung der Darstellungen mit *srcorrp* ist der Parameterbereich auf die Intervalle [0.004;0.006] für $b(1)$ und [0.02;0.034] für $b(2)$ begrenzt und ein Gitter mit 10000 Gitterpunkten eingesetzt worden.
In Abb. 7-1 erkennt man, daß der konturgetreue gemeinsame Vertrauensbereich ebenfalls einer Ellipse ähnelt, aber der Parameterschätzpunkt nicht mehr im Zentrum dieses Vertrauensbereichs liegt.
Auf eine graphische Gegenüberstellung der entsprechenden Vertrauensbereiche aus den verschiedenen Rechengängen ist verzichtet worden, da sie sich nur geringfügig voneinander unterscheiden.

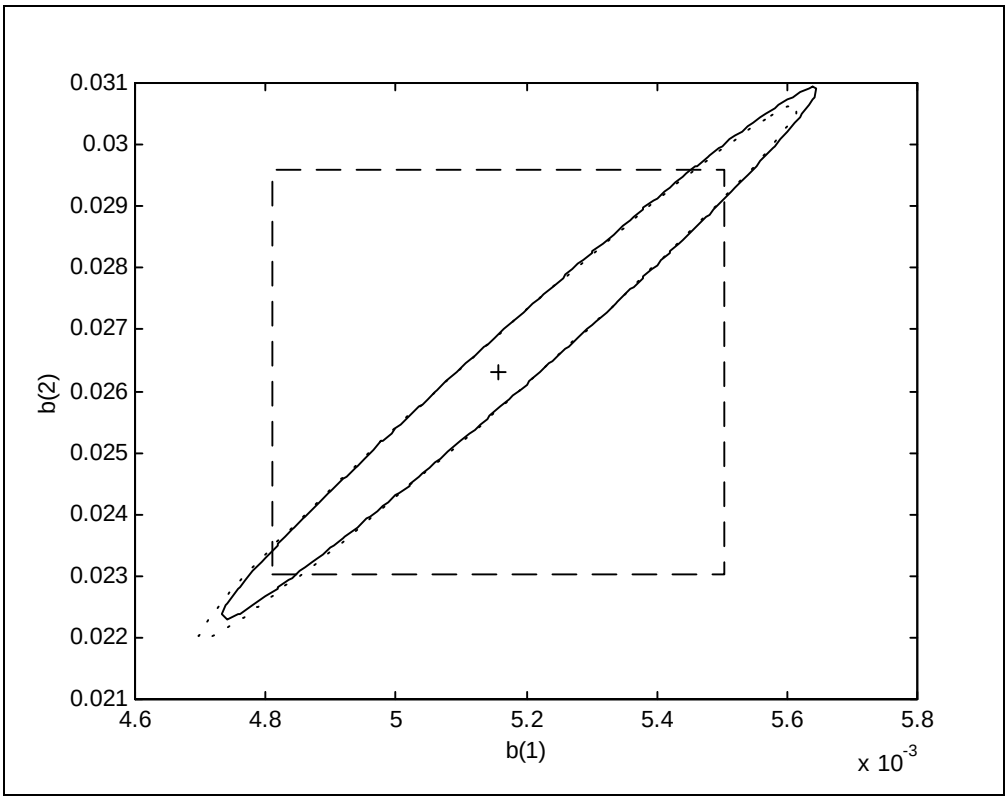

Abb. 7-1. Darstellung verschiedener Vertrauensbereiche für die Parameter einer Hydrierreaktion bei Schätzung mit *singleres* (-- individueller linearisierter Konfidenzbereich (95%), ··· gemeinsame Vertrauensbereichsellipse, – konturgetreuer gemeinsamer Vertrauensbereich, + Parameterschätzpunkt)

7.6 Spezialfall: Arrhenius-Gleichung

Verzeichnis: *arrhenius*
Funktionen: *arrhenius.m*
arcorrp.m

Wegen der großen Bedeutung für die chemische Kinetik und ihrem hohen Maß an Nichtlinearität [9,38] sind in der Programmsammlung spezielle Funktionen zur Parameterschätzung und Parameteranalyse der Arrhenius-Gleichung entwickelt worden. Die Arrhenius-Gleichung

$$k = k_0 \cdot \exp\left(-\frac{E_a}{R \cdot T}\right) \tag{7-16}$$

gibt die Temperaturabhängigkeit der Geschwindigkeitskonstanten k einer chemischen Reaktion an. Die entscheidenden Parameter sind die Aktivierungs-

energie E_a und der Häufigkeitsfaktor k_0. Beide Parameter werden hochkorreliert gefunden und erschweren deshalb eine präzise Parameterschätzung [38,39].

Tab. 7-8. Eigenschaften der Funktionen *arrhenius und arcorrp*

arrhenius:	Bestimmung der Arrhenius-Parameter
Eingabe:	Temperaturvektor, Vektor der Geschwindigkeitskonstanten, Modellwahl*, Optimieroptionen*, Optimierverfahren*, Konfidenzzahl*
Ausgabe:	Schätzwert des Häufigkeitsfaktors, Schätzwert der Aktivierungsenergie, jeweils zwei Vektoren mit den individuellen Vertrauensintervallhalblängen des Häufigkeitsfaktors und der Aktivierungsenergie (Standardabweichung, Konfidenzintervall, Tangentialebenenintervall), Fehlervarianz, Varianz-Kovarianz-Matrix der Schätzung, Fehlerquadratsumme, Modellgleichung, Startwerte, Fehlerquadratsumme zu den Startwerten
arcorrp:	Darstellung des konturgetreuen gemeinsamen Vertrauensbereichs von den Arrhenius-Parametern
Eingabe:	Temperaturvektor, Vektor der Geschwindigkeitskonstanten, Schätzvektor der Arrhenius-Parameter, Modellgleichung, Fehlervarianz, graph. darzustellende Parameter, zulässiger Parameterbereich*, Anzahl der Gitterpunkte je Achse*, Konfidenzzahl*
Ausgabe:	Matrix der Niveaupunkte, Parameterwerte

* = Eingabe freigestellt

Die Schätzung der Arrhenius-Parameter erfolgt mittels der Funktion *arrhenius* (vgl. Tab. 7-8). Wegen des spezifischen Charakters der Funktion ist die Vorgabe einer Modellgleichung nicht notwendig. Ferner werden im Gegensatz zu den bisherigen Methoden programmintern die Startwerte der nichtlinearen Regression als Ergebnis einer linearen Regression mit *linreg* auf der Basis der Logarithmierung von Gl. (7-16) erhalten (vgl. Kap. 6.1). Dafür werden die Temperaturwerte programmintern geeignet skaliert. Die nichtlineare Regression kann dann optional direkt an der Arrhenius-Gleichung oder an ihrer reparametrisierten Form

$$k = k_0 \cdot \exp\left(-\frac{E_a}{R \cdot \overline{T}}\right) \cdot \exp\left(-\frac{E_a}{R} \cdot \left(\frac{1}{T} - \frac{1}{\overline{T}}\right)\right) = k_0' \cdot \exp\left(-\frac{E_a}{R} \cdot \left(\frac{1}{T} - \frac{1}{\overline{T}}\right)\right) \qquad (7\text{-}17)$$

durchgeführt werden. Der Vorteil der Reparametrisierung besteht darin, daß die Konvergenzgeschwindigkeit des Verfahrens erhöht wird und die zu schätzenden Parameter k_0' und E_a eine ähnliche Größenordnung haben, während k_0 und E_a sich durchaus um einen Faktor 10^{10} unterscheiden können. Dieser Größenunterschied

ist für die MATLAB-Optimierverfahren, bei denen Optionseinstellungen, wie z. B. Schrittweiten, für alle Parameter gleichermaßen gelten, besonders schwierig zu handhaben [9,40].
Zur eigentlichen Optimierung und Berechnung der Fehlervarianz wird intern *singleres* angewendet (vgl. Kap. 7.3). Die Schätzung erfolgt bezüglich der ungewichteten Fehlerquadratsumme (vgl. Tab. 7-4).
Neben den Schätzwerten für die Arrhenius-Parameter erhält man über *arrhenius* die individuellen Vertrauensintervalle und die Varianz-Kovarianz-Matrix. Die dazu notwendige Jacobi-Matrix wird über die programminterne Anwendung der Funktion *lininp* erhalten (vgl. Kap. 7.1). Die individuellen Vertrauensintervalle selbst werden mit *lpconfp* bestimmt (vgl. Kap 7.2).
Auf eine Besonderheit bei *arrhenius* muß noch hingewiesen werden. Bei einer Parameterschätzung über Gl. (7-17) wird der Häufigkeitsfaktor als Vektor ausgegeben, dessen erster Wert k_0 und zweiter k_0' ist.

Insgesamt besteht die Aufgabe von *arrhenius* in der Modellformulierung, der Startwertermittlung und gegebenenfalls der Transformation und Rücktransformation der Ergebnisse sowie der Koordinierung und Anwendung diverser Funktionen dieser Programmsammlung. Die Funktion *arrhenius* stellt somit ein Beispiel für die grundlegenden Eigenschaften und die Anwendbarkeit der bisher vorgestellten Programme dar. Zusätzlich zeigt diese Funktion den Einsatz dieser Programme als effiziente Programmierbausteine und verdeutlicht deren Kombinierbarkeit untereinander.

Für die Darstellung der Korrelation der Arrhenius-Parameter steht die Funktion *arcorrp* zur Verfügung. Die Verwendung und der Aufbau von *arcorrp* ist analog zu *srcorrp* (vgl. Kap. 7.4). Zwischen beiden Routinen bestehen lediglich Detailunterschiede aufgrund der hier speziell betrachteten Arrhenius-Gleichung.
Andere Vertrauensbereiche wie beispielsweise das Konfidenzellipsoid können unter Vorgabe der Modellgleichung mit *lpcorrp* oder *lincorrp* berechnet werden (vgl. Kap. 6.1 u. 7.2).

7.7 Anwendungsbeispiel zur Parameterschätzung in der Arrhenius-Gleichung

Die Einsatzfähigkeit dieser Programme soll an einem Beispiel nach [38] demonstriert werden. Dabei ist die katalysierte Reaktion von Ethanol mit Essigsäure betrachtet worden.

Die bei verschiedenen Temperaturen ermittelten Geschwindigkeitskonstanten sind in Tab. 7-9 aufgeführt.

Tab. 7-9. Experimentelle Daten für die katalytische Reaktion zwischen Ethanol und Essigsäure (T Temperatur, k Geschwindigkeitskonstante)

T (in °C)	k
30	0.5
40	1.1
50	2.2
60	4.0
70	6.0

Als Optimierverfahren für die Parameterschätzung ist *fmins* mit Standardeinstellungen gewählt worden. Die Parameterschätzung ist sowohl auf der Grundlage von Gl. (7-16) als auch von Gl. (7-17) durchgeführt und die gemeinsamen Vertrauensbereiche sind mit *arcorrp* berechnet worden.

Die Ergebnisse sind denen der Literatur in Tab. 7-10 gegenübergestellt. Die Literaturergebnisse sind mittels einer nichtlinearen Regression über ein Newton-Raphson-Verfahren ermittelt worden. Zusätzlich sind die Ergebnisse der linearen Regression nach einer Logarithmierung von Gl. (7-15) in die Betrachtungen einbezogen worden.

Tab.7-10. Ergebnisse verschiedener Schätzmethoden von Arrhenius-Parametern bei einer katalysierten Reaktion von Ethanol mit Essigsäure (k_0 Häufigkeitsfaktor, E_a Aktivierungsenergie, s_k, s_E Standardabweichungen, SSR Residuenquadratsumme, s_U^2 Schätzwert der Fehlervarianz)

	arrhenius			Literatur [38]	
	Log-linear	Nichtlinear	Reparam.	Log-linear	Nichtlinear
k_0	1.238*e*9	1.0410*e*8	1.0410*e*8	1.2416*e*9	1.0437*e*8
E_a (in J)	5.4339*e*4	4.7499*e*4	4.7499*e*4	5.4338*e*4	4.7499*e*4
s_k	–	1.2043*e*8	0.6253*e*7	–	–
s_E	–	0.3248*e*4	0.3248*e*4	–	–
SSR	0.260	0.1496	0.1496	0.4893	0.1496
s_U^2	–	0.0499	0.0499	–	–

Die mit *arrhenius* erzielten Ergebnisse entsprechen durchaus den Literaturergebnissen, wobei bei der Betrachtung der Fehlerquadratsumme die hier berechneten Werte im linearen Fall eine höhere Genauigkeit aufweisen.
Die über nichtlineare Regression geschätzten Parameter stimmen gut überein, obwohl sie mit unterschiedlichen Verfahren ermittelt worden sind. Dies kann als Hinweis betrachtet werden, daß es sich dabei tatsächlich um die optimalen Parameter handelt.
Der Vorteil der Reparametrisierung zeigt sich vor allem bei den erhaltenen Parameterwerten in den Standardabweichungen des Häufigkeitsfaktors, der über dieses Verfahren um einen Faktor 20 genauer bestimmt werden kann. Die Ursache hierfür ist die bereits angesprochene Korrelation der Arrhenius-Parameter (vgl. Kap. 7.6).

Die mit *arcorrp* berechneten gemeinsamen Vertrauensbereiche sind in Abb. 7-2 und 7-3 graphisch dargestellt.

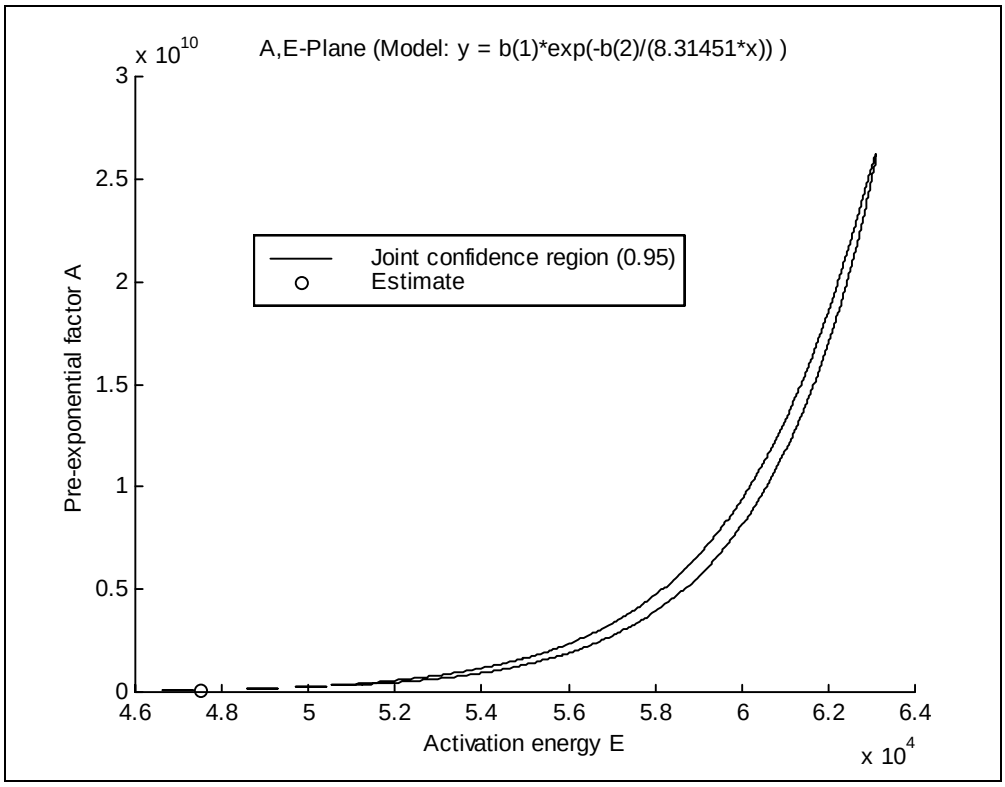

Abb. 7-2. Darstellung des konturgetreuen gemeinsamen Vertrauensbereichs der Arrhenius-Parameter mit *arcorrp*

In Abb. 7-2 ist der konturgetreue gemeinsame Vertrauensbereich für die Arrhenius-Parameter eingezeichnet. Der Punkt der Parameterschätzwerte liegt

im unteren Bereich des Kurvenverlaufs. Der Vertrauensbereich ist ausgesprochen langgestreckt und schmal, was die Parameterschätzung besonders erschwert [38].
In Abb. 7-2 erkennt man auch die Auswirkungen der Korrelation auf die Niveaulinienbestimmung, da es auch bei 350 Gitterpunkten pro Achse nicht möglich gewesen ist, einen vollständig geschlossenen Kurvenverlauf zu erzeugen, und es statt dessen im unteren Bereich zur „Inselbildung“ gekommen ist.

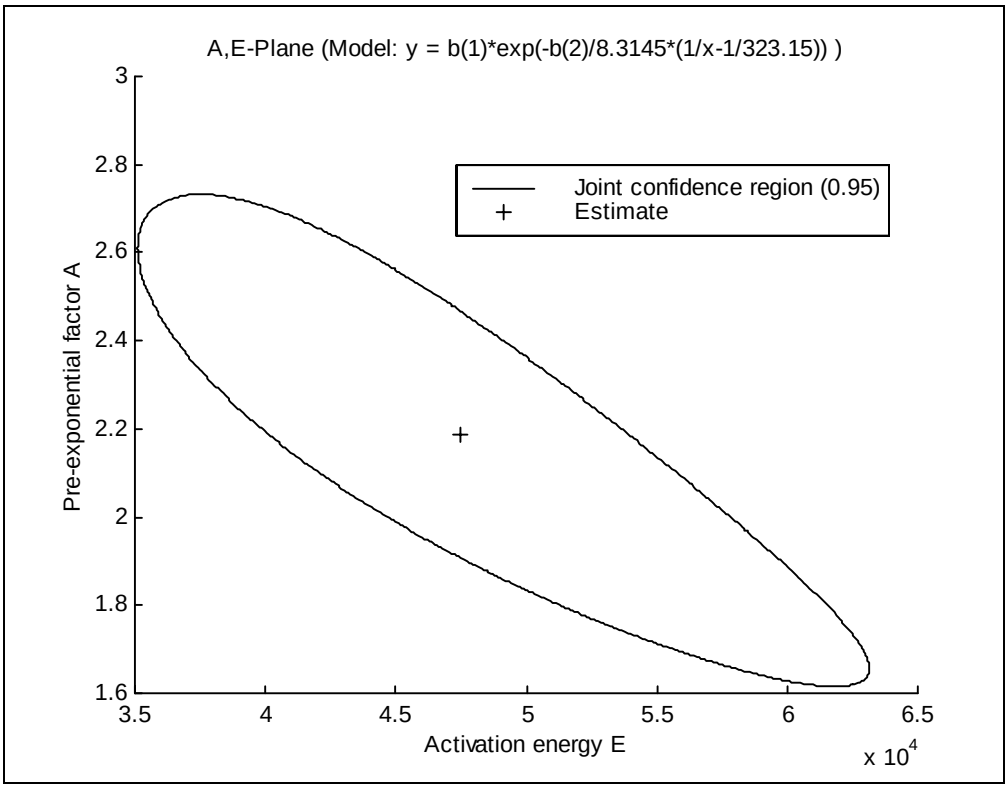

Abb. 7-3. Darstellung des konturgetreuen gemeinsamen Vertrauensbereichs der Parameter der reparametrisierten Arrhenius-Gleichung mit *arcorrp*

Aus Abb. 7-3 wird ersichtlich, daß durch die Reparametrisierung für die zu schätzenden Parameter k_0' und E_a ein viel ausgewogenerer Vertrauensbereich erzielt worden ist, durch den der Einfluß der Korrelation der Parameter auf die Parameterschätzung reduziert werden konnte.

Die erhaltenen Ergebnisse und insbesondere die Darstellungen der Korrelationsbereiche unterstreichen die Leistungsfähigkeit der in MATLAB implementierten Verfahren.

8 Regression bei Mehrfachantwortsystemen

Parameterschätzungen in Mehrfachantwortsystemen sind neben den schon betrachteten Regressionsproblemen bei Einfachantwortsystemen für die Reaktionskinetik von grundlegender Bedeutung, da ein chemisches Reaktionssystem im allgemeinen durch die Beteiligung verschiedener Substanzen und durch den Ablauf unterschiedlicher Reaktionen gekennzeichnet sein kann. Bei der Bestimmung der kinetischen Größen wird man sich deshalb nicht auf den Konzentrations-Zeit-Verlauf einer Komponente beschränken, sondern durch Messung möglichst vieler Komponenten versuchen, das Vertrauen in die Ergebnisse zu erhöhen [41,42]. Dadurch erhält man ein Mehrfachantwortsystem. Bei den Antworten handelt es sich weitgehend um Stoffmengen, Konzentrationen, Geschwindigkeiten o. ä., die sich in der mathematischen Beschreibung explizit darstellen lassen. Aus diesem Grund werden in der vorliegenden Programmsammlung nur Systeme betrachtet, die bezüglich der Antworten $\boldsymbol{\eta}$, der Parameter $\boldsymbol{\beta}$ und der unabhängigen Variablen $\mathbf{x}$ folgenden Zusammenhang an einem Versuchspunkt i aufweisen

$$\boldsymbol{\eta}_i = \mathbf{f}(\boldsymbol{\beta}, \mathbf{x}_i) \tag{8-1}$$

Für die nachfolgende Behandlung von Mehrfachantwortsystemen werden die schon vorgestellten Verfahren für Einfachantworten erweitert und zusätzliche Größen eingeführt. So können beispielsweise die Residuen nicht mehr als Vektor betrachtet werden, sondern stellen sich bei n Versuchspunkten und r Antworten pro Versuch als (r x n)-Matrix $\mathbf{E}$ dar. Damit ergibt sich der Schätzwert der Fehlermatrix $\mathbf{D}$ aus der Momentenmatrix der Residuen in Abhängigkeit des Vektors der Parameterschätzwerte $\mathbf{b}$ zu

$$\mathbf{D}(\mathbf{b}) = \sum_{i=1}^{n} \mathbf{e}_i(\mathbf{b}) \cdot \mathbf{e}_i(\mathbf{b})^T = \mathbf{E}(\mathbf{b}) \cdot \mathbf{E}(\mathbf{b})^T \tag{8-2}$$

Um den unterschiedlichen Einfluß der verschiedenen Antworten auf das System zu berücksichtigen, werden auch die Schätzkriterien erweitert. Sie sind mit ihren Eigenschaften und Anforderungen in Tab. 8-1 aufgeführt und aus [7,9] entnommen.
Bei der Aufstellung der Kriterien sind folgende zwei Voraussetzungen getroffen worden:

- Die Fehler der j-ten Antwort am i-ten Versuchspunkt seien unabhängig und um den Mittelwert Null mit der Varianz σ^2_{ji} normalverteilt.

- Die modellabhängigen Varianzen sollen bei einem Modell für alle Versuchspunkte gleich sein. Die symmetrische Varianz-Kovarianz-Matrix der Meßfehler für verschiedene Antworten hat somit die Dimension (r x r) und unterscheidet sich damit von der Varianz-Kovarianz-Matrix bei Einfachantworten mit der Dimension (n x n).

Tab. 8-1. Schätzfunktionen für Mehrfachantwortsysteme und ihre Beschreibung nach [7,9]

Kriterium	Beschreibung	Schätzfunktion $\Psi(\mathbf{D})$	$\boldsymbol{\Gamma} = \frac{\partial \Psi(\mathbf{D})}{\partial \mathbf{D}}$
Φ_1	Normalverteilung; Varianz-Kovarianz-Matrix bekannt; gewichtete Fehlerquadratsumme	$\sum_{j=1}^{r}\sum_{i=1}^{r} V_{ij}^{-1} \cdot D_{ij}$	$\mathbf{V}^{-1}$
Φ_2	Normalverteilung; Varianzen eventuell bekannt; Kovarianzen 0; gewichtete / ungewichtete Fehlerquadratsumme; für fehlende Daten geeignet	$tr(\mathbf{V}^{-1} \cdot \mathbf{D})$	$\mathbf{V}^{-1}$
Φ_3	Normalverteilung; Varianz-Kovarianz-Matrix unbekannt; gewichtete Fehlerquadratsumme	$\sum_{j=1}^{r} \ln(D_{jj})$	$\Gamma_{jj} = D_{jj}^{-1}$ $\Gamma_{ij} = 0 \quad , \quad i \neq j$
Φ_4	Normalverteilung; Varianz-Kovarianz-Matrix unbekannt; Determinantenkriterium	$\det(\mathbf{D})$	$\det(\mathbf{D}) \cdot \mathbf{D}^{-1}$
Φ_5	Normalverteilung; Varianzen unbekannt; Kovarianzen 0; Determinantenkriterium für fehlende Daten	$\prod_{j=1}^{r} D_{jj}^{\frac{n_j}{2}}$	$\Gamma_{jj} = \frac{n_j}{2} \cdot \frac{\Psi}{D_{jj}}$ $\Gamma_{ij} = 0 \quad , \quad i \neq j$

Unter Beachtung obiger Vorgaben stellt das Kriterium Φ_1 die Ableitung einer allgemeinen Form der gewichteten Quadratsumme dar. Um dieses Kriterium anwenden zu können, ist eine Schätzung der Varianz-Kovarianz-Matrix vorab notwendig.

Bei dem Kriterium Φ_2 handelt sich um das häufig angewendete Kriterium der Summe der Residuenquadrate. Bei Kenntnis der Varianzen oder eines Vielfaches der Varianzen der einzelnen Modelle können die Summen der Residuenquadrate

der einzelnen Modelle gewichtet werden. Sind alle Varianzen gleich oder als gleich angenommen, entspricht das Kriterium der Summation aller Fehlerquadrate ohne Berücksichtigung der Eigenschaften der verschiedenen Antworten und stellt damit eine sehr starke Vereinfachung dar [9]. Das Kriterium Φ_2 kann ferner bei fehlenden Beobachtungswerten verwendet werden.
Die Kriterien Φ_3 ,Φ_4 und Φ_5 sind von besonderer Bedeutung, da sie die experimentell schwer zugänglichen Varianzen und Kovarianzen nicht als bekannt voraussetzen. Allerdings sind diese Größen indirekt in der Herleitung der Kriterien über ein Maximum-Likelihood-Schätzverfahren berücksichtigt worden. Deshalb lassen sich bei der Optimierung auf diese Kriterien Maximum-Likelihood-Schätzwerte für die Varianz-Kovarianz-Matrix errechnen, die jedoch nicht erwartungstreu sind [9] (vgl. Kap. 2.2). Bei Φ_3 und Φ_5 werden die Kovarianzen zwischen den Antworten noch als Null vorausgesetzt, während dieses bei Φ_4 nicht notwendig ist. In seiner Allgemeinheit ist das von BOX und DRAPER abgeleitete Determinantenkriterium Φ_4 mit dem Kriterium Φ_1 vergleichbar [43].
Für den Fall, daß das Modell nur aus einer Modellgleichung besteht, lassen sich alle Kriterien auf die Methode der kleinsten Fehlerquadrate bei Einfachantworten zurückführen.
Alle in Tab. 8-1 aufgeführten Schätzkriterien sind in der Zentralfunktion *mrobjfcn* abgelegt (vgl. Kap 3.5.2).
Die Auswahl des Schätzkriteriums bei der Parameterschätzung sollte sich vordergründig nach der vorhandenen Datensituation richten, weshalb ein bewertender Vergleich der aufgeführten Methoden untereinander nicht unbedingt sinnvoll ist. Es zeigt sich allerdings nach [9,44], daß sich Φ_4 vor allem gegenüber Φ_2 durch eine schnellere Konvergenz und durch präzisere Parameterschätzwerte auszeichnet.
Ferner wird die Anwendbarkeit der Kriterien durch Abhängigkeiten in den Antworten oder der Fehler beeinflußt [9,45].

8.1 Untersuchung auf lineare Abhängigkeiten in den Daten

Verzeichnis: *multiresponse*
Funktionen: *lindep.m*

Lineare Abhängigkeiten in den Antworten eines Mehrfachantwortsystems können die Anwendbarkeit von Schätzkriterien beeinflussen, weshalb vor einer Parameterschätzung eine Untersuchung auf diese Abhängigkeiten durchgeführt werden sollte.

Nach [9,45] können drei Formen linearer Abhängigkeit bei Mehrfachantworten unterschieden werden:

- Die Abhängigkeit zwischen den Fehlern, d. h. ein positiver Fehler in der einen Antwort bewirkt einen positiven Fehler in einer anderen, was in keinem Schätzkriterium nach Tab. 8-1 berücksichtigt wird.
- Die lineare Abhängigkeit zwischen den Erwartungswerten der Antworten, wie sie beispielsweise aufgrund der stöchiometrischen Beziehungen zwischen Reaktionsteilnehmern vorliegen kann. Diese Form der linearen Abhängigkeit kann bei Parameterschätzungen aufgrund des Vorliegens von zusätzlichen Beziehungen zwischen den Parametern die Genauigkeit der Schätzung erhöhen.
- Die lineare Abhängigkeit zwischen den Antworten, die dazu führt, daß die Anwendung von Schätzkriterien eingeschränkt sein kann, so darf in diesem Fall beispielsweise das Determinantenkriterium nicht eingesetzt werden [9,45].

Zur Untersuchung derartiger Abhängigkeiten wird im Rahmen dieser Programmsammlung die Funktion *lindep* zur Verfügung gestellt (vgl. Tab. 8-2). In *lindep* wird gemäß [9,45] eine empirische Eigenwert-Eigenvektoranalyse durchgeführt. Dazu werden nach Vorgabe der Antwortmatrix **Y** mittels der MATLAB-Funktion *eig* die Eigenwerte und Eigenvektoren der Matrix $\mathbf{\Delta}^T\mathbf{\Delta}$ bestimmt nach

$$\Delta_{ij} = Y_{ji} - \bar{y}_j \qquad \text{und} \qquad \bar{y}_j = \frac{1}{n} \cdot \sum_{i=1}^{n} Y_{ji} \tag{8-3}$$

Ein Eigenwert von Null bedeutet hierbei, daß lineare Abhängigkeiten in den Daten vorhanden sind, während ein Eigenwert in der Größenordnung des (n-1)-fachen Werts der Fehlervarianz auf eine Abhängigkeit unter den Erwartungswerten der Antworten hinweist.

Tab. 8-2. Eigenschaften der Funktion *lindep*

lindep:	Eigenwert-Eigenvektoranalyse zur Untersuchung von linearen Abhängigkeiten
Eingabe:	Matrix
Ausgabe:	Matrix der Eigenvektoren, Eigenwerte

8.2 Parameterschätzung und Bestimmung individueller Vertrauensbereiche

Verzeichnis: *multiresponse*
Funktionen: *multires.m*
mrconfp.m

Die eigentliche Parameterschätzung in Mehrfachantwortsystemen erfolgt über die Routine *multires*. Hierfür müssen die Matrix der unabhängigen Variablen, die Antwortmatrix, die Startschätzwerte für die Parameter und die Modellgleichungen als *cell array* und in ybx-Schreibweise zur Verfügung gestellt werden. Ferner können die Varianz-Kovarianz-Matrix der Antworten, die Optimieroptionen und das Bewertungskriterium vorgegeben bzw. ausgewählt werden. Als Optimierverfahren können wie üblich die Funktionen *fmins*, *leastsq* und *constr* eingesetzt werden (vgl. Kap. 3.5.1). Bei Verwendung von *constr* kann zusätzlich der Parameterraum eingeschränkt werden, oder es können zusätzliche Nebenbedingungen zwischen abhängigen oder unabhängigen Variablen oder den Parametern analog zu den Modellgleichungen implementiert werden. Die Eigenschaften von *multires* sind in Tab. 8-3 zusammengefaßt.

Tab. 8-3. Eigenschaften der Funktionen *multires* und *mrconfp*

multires:	Parameterschätzung in Mehrfachantwortsystemen
Eingabe:	Matrix der unabhängigen Variablen, Antwortmatrix, Varianz-Kovarianz-Matrix der Antworten*, Startschätzwerte, Optimieroptionen*, Optimierverfahren*, Schätzkriterium*, zulässiger Parameterbereich*, Nebenbedingungen*
Ausgabe:	Schätzwert der Parameter, Varianz-Kovarianz-Matrix der Antworten, Wert der Schätzfunktion, Residuen
mrconfp:	Bestimmung individueller linearisierter Vertrauensintervalle von Parametern in nichtlinearen Modellen bei Mehrfachantwortsystemen
Eingabe:	Matrix der unabhängigen Variablen, Antwortmatrix, Schätzvektor der Parameter, Modellgleichungen, gewählte Schätzfunktion*, Varianz-Kovarianz-Matrix der Antworten*
Ausgabe:	Parameterschätzwerte, Varianz-Kovarianz-Matrix der Schätzung, Standardabweichung, Konfidenzintervallhalblängen, Halblängen des Tangentialebenenvertrauensintervalls, Wert der Schätzfunktion, Gradient, Jacobi-Matrix, Hesse-Matrix (jeweils auf die Parameter bezogen)

*= Eingabe freigestellt

Die Routine *multires* koordiniert als übergeordnete Funktion den Informationsfluß zu den Unterprogrammen und beinhaltet den Aufruf des Optimiernavigators *optrout*. Das Bewertungskriterium wird in *mroptfcn* dem Optimierverfahren zur Verfügung gestellt und in *mrobjfcn* berechnet (vgl. Abb. 3-2 u. Kap. 3.5).
Neben den Parameterschätzwerten erhält man gegebenenfalls eine Schätzung für die Varianz-Kovarianz-Matrix der Antworten, die Residuenmatrix und den Wert der Schätzfunktion für den optimalen Parametersatz.

Die Bestimmung der individuellen Vertrauensbereiche der Parameter erfolgt mit der Funktion *mrconfp* (vgl. Tab. 8-3). Durch diese separate Berechnung der Vertrauensintervalle besteht auch hier die Möglichkeit, Konfidenzbereiche von anderweitig ermittelten Parametern zu untersuchen.
Über *mrconfp* werden dabei die Varianz-Kovarianz-Matrix der Schätzung, die Standardabweichungen, die Tangentialebenenvertrauensintervalle, der Wert der Schätzfunktion, der Gradient, die Jacobi-Matrix und die Hesse-Matrix erhalten. Die Argumentenliste von *mrconfp* ist ähnlich zu der von *multires*. In dem Programm selbst werden die Modellwerte der Antworten berechnet und darüber die Residuen und die Fehlermatrix bestimmt. Mit Hilfe der Funktion *jacsym* wird aus den Modellgleichungen die Jacobi-Matrix **J** ermittelt (vgl. Kap. 4.2). Über die Beziehungen gemäß Gl. (7-9) bis Gl. (7-11) und Tab. 8-1 werden die Hesse-Matrix **H**, die Varianz-Kovarianz-Matrix der Schätzung $\mathbf{V}_b$ und der Gradient der Schätzfunktion **q** ermittelt.
In der Berechnung von $\mathbf{V}_b$ nach Gl. (7-11) wird die Varianz-Kovarianz-Matrix der Antworten im Programm jedoch nur berücksichtigt, wenn sie extern vorgegeben wird. Dieses bedeutet, daß der Anwender eine experimentell ermittelte oder durch *multires* geschätzte Matrix verwenden kann und sich nicht an der Wahl der ursprünglichen Schätzfunktion orientieren muß. Über die Varianz-Kovarianz-Matrix der Schätzung werden dann analog zur Funktion *srconfp* die individuellen Vertrauensintervalle ermittelt (vgl. Kap. 7.4).

8.3 Gemeinsame Vertrauensbereiche der Parameter

Verzeichnis: *multiresponse*
Funktion: *mrcorrp1.m*
mrcorrp2.m
mrcorrp4.m
mrcorrp5.m

Bei einer Parameterschätzung mit einem aus Tab. 8-1 gewählten Schätzkriterium können außer für Φ_3 mit den Funktionen *mrcorrp1*, *mrcorrp2*, *mrcorrp4* und *mrcorrp5* jeweils paarweise die Korrelationen zwischen den Parametern untersucht werden. Die Ziffern im Funktionsnamen entsprechen hierbei dem Bewertungskriterium nach Tab. 8-1. Der Programmaufbau und -ablauf ist dabei ähnlich der Routine *srcorrp* (vgl. Kap. 7.4). Über die Funktionen *mrcorrp1-5* erhält man auch die notwendigen Daten zur Berechnung der konturgetreuen Vertrauensintervalle (vgl. Kap. 7.4). Die Eigenschaften dieser Funktionen sind in Tab. 8-4 aufgeführt.

Tab. 8-4. Eigenschaften der Funktionen zur Bestimmung gemeinsamer Vertrauensbereiche bei Mehrfachantwortsystemen

Darstellung des konturgetreuen gemeinsamen Vertrauensbereichs von zwei ausgewählten Parametern in nichtlinearen Modellen bei Mehrfachantwortsystemen	
mrcorrp1:	Schätzfunktion: allgemeine gewichtete Residuenquadratsumme
mrcorrp2:	Schätzfunktion: gewichtete oder ungewichtete Residuenquadratsumme
mrcorrp4:	Schätzfunktion: Determinantenkriterium
mrcorrp5:	Schätzfunktion: modifiziertes Determinantenkriterium für fehlende Daten
Eingabe:	Matrix der unabhängigen Variablen, Antwortmatrix, Schätzvektor der Parameter, Modellgleichungen, Varianz-Kovarianz-Matrix der Antworten**, graphisch darzustellende Parameter, zulässiger Parameterbereich*, Anzahl der Gitterpunkte je Achse*, Konfidenzzahl*, graphische Darstellung (j/n)*
Ausgabe:	Matrix der Niveaupunkte, Parameterwerte

*= Eingabe freigestellt, ** = nur bei *mrcorrp1/2*

Die Grundlage für die Berechnung der Niveaulinien stellt in *mrcorrp1* und *mrcorrp2* Gl. (6-9) dar.
Für die konturgetreue Darstellung bei Anwendung der Determinantenkriterien werden analoge Beziehungen verwendet. Für Φ_4 gilt dabei nach BOX und DRAPER [44,45]

$$\left(\Phi_4\right)_\gamma = \left(\det(\mathbf{D})\right)_\gamma = \left(\det(\mathbf{D})\right)_{Min} \cdot \exp\left(\frac{\chi_p^2(\gamma)}{n}\right). \qquad (8\text{-}4)$$

Der Zähler im Argument der Exponentialfunktion stellt den Abszissenwert dar, an dem die Fläche unter der χ^2-Verteilungskurve (γ x 100) % beträgt. Dieser Wert ist abhängig von den Freiheitsgraden und der Anzahl der Parameter.

Für das abgewandelte Determinantenkriterium Φ_5 , für das die Kovarianzen der Antworten als Null angenommen werden, lautet die Beziehung

$$\left(\Phi_5\right)_\gamma = \prod_{i=1}^{r} \left(D_{ii}\right)_{Min}^{\frac{n_i}{2}} \cdot \exp\left(\frac{\mathrm{X}_p^2(\gamma)}{2}\right) \tag{8-5}$$

Eine Herleitung von Gl. (8-5) befindet sich in Kap. 14, Anhang A.
Die Berechnung des X^2-Wertes erfolgt in den Funktionen *mrcorrp4* und *mrcorrp5* über die Funktion *chi2distinv* (vgl. Kap. 4.3).

8.4 Anwendungsbeispiel

Die in diesem Kapitel vorgestellten Programme sollen an einem einfachen Reaktionsnetzwerk auf ihre Einsatzfähigkeit untersucht. Das Beispiel ist [9] entnommen und stellt ein Simulationsexperiment zu nachstehender Folgereaktion dar.

$$\mathrm{A} \xrightarrow{k_1} \mathrm{B} \xrightarrow{k_2} \mathrm{C} \tag{8-6}$$

Hierbei gilt es, die Geschwindigkeitskonstanten k_1 und k_2 zu bestimmen.
Für die zeitliche Abhängigkeit der Konzentrationen gelte für $t > 0$

$$\begin{aligned} c_A &= \exp(-k_1 \cdot t) \\ c_B &= \frac{k_1}{k_2 - k_1} \cdot \left(\exp(-k_1 \cdot t) - \exp(-k_2 \cdot t)\right) \\ c_C &= 1 - c_A - c_B \end{aligned} \tag{8-7}$$

bzw. in programmgerechter ybx-Schreibweise mit t als unabhängiger Variable x

$$\begin{aligned} y(1) &= \exp\left(-b(1) \cdot x\right) \\ y(2) &= \frac{b(1)}{b(2) - b(1)} \cdot \left[\exp\left(-b(1) \cdot x\right) - \exp\left(-b(2) \cdot x\right)\right] \\ y(3) &= 1 - \exp\left(-b(1) \cdot x\right) - \frac{b(1)}{b(2) - b(1)} \cdot \left[\exp\left(-b(1) \cdot x\right) - \exp\left(-b(2) \cdot x\right)\right] \end{aligned} \tag{8-8}$$

Die Anfangskonzentrationen zum Zeitpunkt $x = t = 0$ seien

$$y_0(1) = c_{A,0} = 1 \qquad \text{und} \qquad y_0(2) = c_{B,0} = y_0(3) = c_{C,0} = 0 \tag{8-9}$$

Die simulierten Meßdaten zu diesem Rechnerexperiment sind in Tab. 8-5 aufgeführt.

Tab. 8-5. Simulierte Daten für die Folgereaktion A → B → C nach [9]

x	$y(1)$	$y(2)$	$y(3)$
0.5	0.939	-0.026	-0.003
0.5	0.891	0.081	0.085
1.0	0.803	0.182	0.091
1.0	0.829	0.134	0.038
2.0	0.696*	0.255	0.115
2.0	0.635*	0.131	0.103
4.0	0.432*	0.198	0.339
4.0	0.450*	0.151	0.332
8.0	0.197*	0.134*	0.725
8.0	0.162*	0.080*	0.634
16.0	0.000*	0.085*	0.940
16.0	0.026*	0.034*	0.923

Bei der Berechnung der Meßdaten nach Gl. (8-7) bzw. Gl. (8-8) sind die Geschwindigkeitskonstanten mit

$$b(1) = k_1 = 0.2080 \qquad b(2) = k_2 = 0.4931 \tag{8-10}$$

und die Varianzen der Antworten zu

$$\sigma_{11}{}^2 = 0.001 \qquad \sigma_{22}{}^2 = 0.0025 \qquad \sigma_{33}{}^2 = 0.001 \tag{8-11}$$

angenommen worden.
Vorgehensweisen und Strategien zur Auswertung des Simulationsexperiments entstammen [9,41]. Nachfolgende Parameterschätzung ist sowohl unter Berücksichtigung aller Meßdaten als auch unter Vernachlässigung der in Tab. 8-5 mit „*“ gekennzeichneten Daten durchgeführt worden.

Voruntersuchung auf lineare Abhängigkeiten
Um die Anwendbarkeit der einzelnen Schätzkriterien zu klären, sind vor der eigentlichen Parameterschätzung die Antworten auf lineare Abhängigkeiten untersucht worden. Dazu ist die Matrix der Antworten gemäß Tab. 8-4 aufgestellt worden. Für die Eigenwert-Eigenvektoranalyse ist *lindep* eingesetzt worden (vgl. Kap. 8.1). Die erhaltenen Eigenvektoren und Eigenwerte sind in Tab. 8-6 aufgeführt.

Tab. 8-6. Ergebnisse der Eigenwert-Eigenvektoranalyse mit *lindep*

Eigenwerte	Eigenvektoren
0.0884	[-0.3757 0.8753 -0.3046]
0.0204	[0.6136 0.4812 0.6261]
2.6321	[-0.6945 -0.0483 0.7178]

Wie erwartet ist kein Eigenwert Null, da alle Konzentrationswerte in Tab. 8-5 als Meßwerte simuliert worden sind. Somit liegt keine lineare Abhängigkeit in den Daten der Antworten vor, und das Determinantenkriterium kann angewendet werden.
Für eine weitere Beurteilung der Ergebnisse in Tab. 8-6 ist die Kenntnis der durchschnittlichen Fehlervarianz notwendig (vgl. Kap. 8.1). Hierbei kann sie entweder aus den Modellgleichungen und Meßdaten abgeschätzt werden oder ist vorab bekannt. Hier ergibt sie sich nach Gl. (8-11) zu

$$\overline{\sigma^2} = 0.0015 \tag{8-12}$$

Für das Vorliegen einer Abhängigkeit in den Erwartungswerten der Antworten müßte nach [45] ein Eigenwert in der Größenordnung von

$$(n-1)\cdot\overline{\sigma^2} = 0.0165 \tag{8-13}$$

liegen. Dies trifft angenähert auf den zweiten Eigenwert zu.

Parameterschätzung und Auswertung
Die Parameterschätzung erfolgte mit *multires*. Als Schätzfunktionen sind die reine Fehlerquadratsumme (Φ_2), das Determinantenkriterium (Φ_4) und das abgewandelte Determinantenkriterium (Φ_5) verwendet worden. Für die Parameterschätzung bei fehlenden Daten sind die Kriterien Φ_2 und Φ_5 angewendet worden, wobei die fehlenden Werte in der Antwortmatrix durch den MATLAB-Ausdruck *NaN* ersetzt worden sind.
Als Optimierverfahren ist *fmins* mit den MATLAB-Standardeinstellungen bei folgenden Startschätzwerten

$$b_0(1) = 1 \qquad b_0(2) = 0.9 \tag{8-14}$$

gewählt worden.

Die Standardabweichungen der Parameter sind mit *mrconfp* berechnet und die Korrelationsdarstellungen mit *mrcorrp2*, *mrcorrp4* und *mrcorrp5* ermittelt worden.
Die aufgrund der Parameterschätzung erhaltenen Ergebnisse sind in Tab. 8-7 aufgeführt.

Tab. 8-7. Parameterschätzwerte nach verschiedenen Schätzverfahren

Nr.	Kriterium	b(1)		b(2)		Φ-Wert
1	Φ_2 ungewichtet alle Werte	0.2004	±0.1848	0.5326	±1.1319	0.0541
2	Φ_2 gewichtet alle Werte	0.2031	±0.0065	0.5062	±0.0419	34.1933
3	Φ_4 alle Werte	0.2064	±0.0053	0.5067	±0.0349	2.5184*e*-6
4	Φ_5 alle Werte	0.2054	±0.0052	0.4997	±0.0421	9.09*e*-34
5	Φ_2 ungewichtet fehlende Werte	0.1900	±0.1750	0.5508	±1.2243	0.0384
6	Φ_2 gewichtet fehlende Werte	0.1917	±0.0061	0.5391	±0.0474	24.6296
7	Φ_5 fehlende Werte	0.1932	±0.0044	0.5336	±0.0472	5.57*e*-24

Diskussion und Beurteilung
Die in Tab. 8-7 aufgeführten Ergebnisse spiegeln unterschiedliche Aspekte wider, die hier ausführlich diskutiert werden sollen.
Die Ergebnisse zeigen eine gute Übereinstimmung mit den erwarteten Parameterwerten gemäß Gl. (8-10). Der Fehlerbereich aller Schätzergebnisse schließt die wahren Werte ein, wobei die durchschnittliche Abweichung bei 5.4 % liegt. Die beste Übereinstimmung ist mit Verfahren Nr. 4 in Tab. 8-7 unter Anwendung des abgewandelten Determinantenkriteriums (Φ_5) auf alle Meßwerte mit einer auf beide Parameter bezogenen durchschnittlichen Abweichung von den wahren Werten von 1.3 % erzielt worden. Insgesamt erweisen sich in dem vorliegenden Beispiel die Determinantenkriterien als die besseren Schätzfunktionen.
Neben diesen Übereinstimmungen zu den wahren Werten zeigen auch die veröffentlichten Schätzergebnisse [9] bei vergleichbaren Auswertungsvorschriften eine sehr geringe Abweichung zu den hier vorgestellten Ergebnissen. Die Ab-

weichung liegt in der Größenordnung von 1 %. Die Ursache für diese Unterschiede in den Ergebnissen ist vor allem im Auftreten von Rundungsfehlern und in den verschiedenen eingesetzten Optimierverfahren zu sehen. Die Literaturergebnisse sind in Tab. 8-8 den entsprechenden Ergebnissen aus Tab. 8-7 gegenübergestellt.

Tab. 8-8. Parameterschätzwerte nach verschiedenen Schätzverfahren gemäß [9] im Vergleich zu den Ergebnissen aus Tab. 8-7

	Ergebnisse nach [9]		Ergebnisse nach Tab. 8-7	
Schätzvorschrift	$b(1)$	$b(2)$	$b(1)$	b(2)
vergleichbar mit Φ_4 alle Werte	0.2062	0.5061	0.2064	0.5067
vergleichbar mit Φ_5 alle Werte	0.2054	0.4991	0.2054	0.4997
vergleichbar mit Φ_5 fehlende Werte	0.1953	0.5412	0.1932	0.5336
wahre Werte	0.2080	0.4931	0.2080	0.4931

Auch ein Vergleich der Ergebnisse untereinander entspricht weitgehend den Erwartungen, da die Schätzwerte in derselben Größenordnung liegen. Anhand der Fehlerbereiche wird jedoch deutlich, daß die Verwendung der ungewichteten Fehlerquadratsumme (Φ_2) als Schätzfunktion zu relativ ungenauen Parametern führt. Die Ursache hierfür liegt in den fehlenden Informationen über die Fehlerstruktur der Antworten. Dieses spiegelt sich auch in den konturgetreuen gemeinsamen Vertrauensbereichen wider, die in Abb. 8-1 und 8-2 für eine Wahrscheinlichkeit von 95 % dargestellt sind.

Vergleicht man in den Abbildungen die Vertrauensbereiche bei Anwendung der ungewichteten Fehlerquadratsumme als Schätzkriterium mit denen der gewichteten Fehlerquadratsumme, so führt der Einsatz des gewichteten Kriteriums erwartungsgemäß zu engeren Vertrauensbereichen.
In den Abbildungen zeigt sich aber auch, daß die Anwendung der Determinantenkriterien zu deutlich kleineren Vertrauensbereichen gegenüber den Quadratsummenkriterien führt, was durch [46] bestätigt wird und nochmals den Vorteil dieser Kriterien unterstreicht.

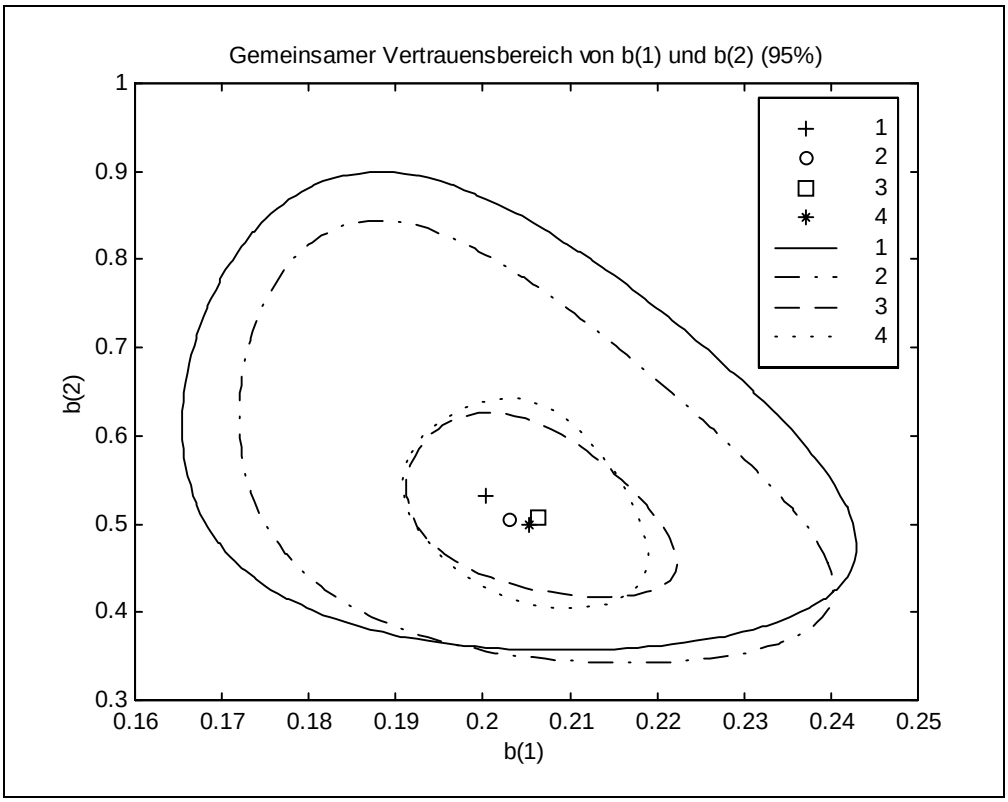

Abb. 8-1. Darstellung konturgetreuer gemeinsamer Vertrauensbereiche für die Schätzvorschriften 1-4 in Tab. 8-7 (Linie: Vertrauensbereich, Symbol: Schätzwert)

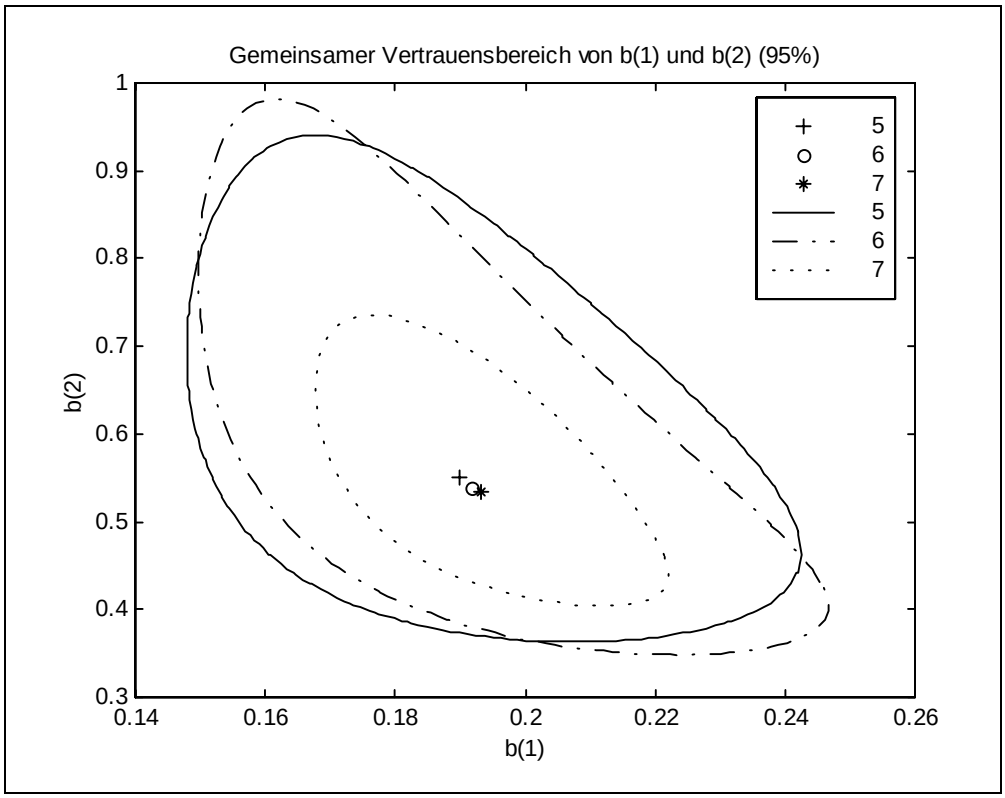

Abb. 8-2. Darstellung konturgetreuer gemeinsamer Vertrauensbereiche für die Schätzvorschriften 5-7 in Tab. 8-7 (Linie: Vertrauensbereich, Symbol: Schätzwert)

In Abb. 8-1 unterscheiden sich ferner die Vertrauensbereiche der Determinantenkriterien nur unwesentlich voneinander, da die Kovarianzen der Antworten gegenüber den Varianzen in diesem Beispiel vernachlässigbar klein sind und somit kein maßgeblicher Unterschied in der Verwendung dieser Kriterien auftritt.
Die in Abb. 8-2 dargestellten gemeinsamen Vertrauensbreiche der Parameterschätzwerte bei fehlenden Meßdaten sind gegenüber ihren Entsprechungen bei Berücksichtigung aller Werte erwartungsgemäß größer gefaßt, weil mit weniger Werten weniger Informationen zur Verfügung stehen.
Die hier erhaltenen gemeinsamen Vertrauensbereiche stimmen ebenfalls mit denen in [9] veröffentlichten gut überein.

Insgesamt läßt sich damit feststellen, daß die gute Übereinstimmung mit den Literaturangaben nicht nur auf die korrekte Umsetzung der Kriterien und Programme hinweist, sondern auch für die Leistungsfähigkeit der MATLAB-Funktion *fmins* als Optimierverfahren spricht.

9 Methoden zur Modelldiskriminierung

Bei den bisher durchgeführten Parameterschätzungen ist die Richtigkeit des zugrunde gelegten Modells vorausgesetzt worden. Allerdings können in der chemischen Kinetik häufig verschiedene Modellannahmen als plausible Beschreibung eines Reaktionsablaufs angesehen werden. Die Aufgabe der Modelldiskriminierung ist es, zu klären, welches dieser Modelle das am besten geeignete ist [47]. Dabei kann folgende Strategie eingesetzt werden:

1. *Vorauswahl*
   Können Parameterschätzungen auf der Grundlage bereits vorhandener Meßdaten ausgeführt werden, so muß die Güte der Parameterschätzung in der Größenordnung des Meßfehlers liegen. Andernfalls kann das Modell verworfen werden. Ist der Meßfehler nicht bekannt, kann eine derartige Vorauswahl nicht getroffen werden.
2. *Modelldiskriminierung*
   Messungen und Versuche werden derart geplant, daß falsche Modelle möglichst schnell eliminiert werden können.

Der erste Schritt kann teilweise mit den bereits vorgestellten Verfahren zur Parameterschätzung umgesetzt werden.
Zur Durchführung der Modelldiskriminierung wird in diesem Kapitel die kanonische Analyse zur Antwortflächenerforschung vorgestellt, die es ermöglicht, durch Analyse empirischer Regressionsgleichungen eine große Zahl postulierbarer mechanistischer Modelle auszuscheiden.
Zudem werden Methoden behandelt, die eine schärfere Unterscheidung verbleibender konkurrierender Modelle erlauben. Anschließend wird auf die Aufstellung von Versuchsplänen eingegangen, die zur Modellunterscheidung bzw. zur simultanen Modellunterscheidung und Parameterschätzung herangezogen werden können, wobei der statistische Aufwand zunehmend höher wird [9].

9.1 Kanonische Analyse zur Antwortflächenerforschung

Verzeichnis: *mdiscrimination*
Funktionen: *mdcanan.m*

Für eine Modellgleichung der allgemeinen Form

$$\eta = f(\boldsymbol{\beta}, \mathbf{x}) \tag{9-1}$$

ergibt die Auftragung der Modellantworten η in Abhängigkeit von m unabhängigen Variablen x und dem Parametervektor β eine m-dimensionale Antwortfläche. Die Aufgabe der Antwortflächenanalyse besteht darin, die Beschaffenheit dieser Antwortfläche in der Nähe eines Extremums zu bestimmen, wodurch

- ein empirisches Modell in Form eines Polynoms zweiten Grades als Approximation an die wahre Antwortfläche,
- Hinweise über physikalisch begründete Modelle und
- Hinweise auf Gebiete, in denen weitere Experimente durchgeführt werden müssen,

erhalten werden [9,48].
Dabei geht man davon aus, daß die Modellfläche durch eine Polynomgleichung zweiten Grades beschrieben werden kann. Diese Gleichung ergibt sich aus einer Taylor-Entwicklung der Modellgleichung, die nach dem zweiten Glied abgebrochen wird. Die Regressionsgleichung lautet dann

$$\hat{y} = b_o + \mathbf{b}^T \cdot \mathbf{x} + \mathbf{x}^T \cdot \mathbf{B} \cdot \mathbf{x} . \qquad (9\text{-}2)$$

In der Matrix **B** sind die Koeffizienten der gemischten und quadratischen Glieder der Polynomgleichung aufgeführt. Diese Matrix ist symmetrisch, denn gewöhnlich werden die gemischten Glieder in die jeweiligen Terme $x_i x_j$ und $x_j x_i$ zerlegt und die ursprünglichen Koeffizienten dafür halbiert und entsprechend in die Matrix **B** eingetragen.
Für eine einfachere Interpretation der Antwortfläche wird diese Regressionsgleichung in die kanonische Form transformiert. Dazu wird der Koordinatenursprung in das Extremum der Antwortfläche überführt. Anschließend folgt eine Rotation der Achsen derart, daß sie den Hauptachsen der Antwortflächenkonturen entsprechen (orthogonale Transformation). Damit ergibt sich die Regressionsgleichung im neuen Koordinatensystem zu

$$\hat{y} - \hat{y}^* = \tilde{\mathbf{x}}^T \cdot \mathbf{M}^T \cdot \mathbf{B} \cdot \mathbf{M} \cdot \tilde{\mathbf{x}} = \lambda_1 \cdot \tilde{x}_1^{\,2} + \lambda_2 \cdot \tilde{x}_2^{\,2} + \ldots + \lambda_m \cdot \tilde{x}_m^{\,2} \qquad (9\text{-}3)$$

Hierbei bezieht sich „~“ auf die Transformationen und „*“ auf das Extremum der Antwortfläche. Die Matrix **M** besteht aus den normierten Eigenvektoren von **B**, deren Eigenwerte λ_1 bis λ_m sind.
Die Eigenwerte der Koeffizientenmatrix **B** lassen folgende Interpretationen der Antwortfläche zu:

1. Alle Eigenwerte haben dasselbe Vorzeichen. In diesem Fall liegen elliptische, also geschlossene, Konturverläufe der Antwortfläche vor.

2. Die Eigenwerte haben unterschiedliche Vorzeichen. Dies deutet auf hyperbolische Konturen hin. Die Antwortfläche verfügt damit über einen Sattelpunkt.
3. Mindestens ein Eigenwert ist Null, wodurch gilt: $\det(\mathbf{B}) = 0$. Hierbei liegen degenerierte Antwortflächen vor, die entlang der entsprechenden Achsen im Hauptachsenraum ins Unendliche gestreckt sind.

Modelle, die in ihrer geometrischen Betrachtung diesen Interpretationen nicht genügen, können als nicht zulässig angesehen werden.

Die Umsetzung der kanonischen Analyse zur Antwortflächenerforschung erfolgt in der vorliegenden Arbeit über die Funktion *mdcanan* (vgl. Tab.9-1). In dieser Routine werden die Koeffizienten der Regressionsgleichung Gl. (9-2) über die Funktion *mlinreg* bestimmt (vgl. Kap. 6.1). Anschließend wird die Matrix **B** aufgestellt und ihre Eigenwerte und Eigenvektoren über die MATLAB-Funktion *eig* berechnet. Im MATLAB-Fenster werden die Interpretationsergebnisse bezüglich der Antwortfläche ausgegeben. Ferner können über die Funktion *mdcanan* die Eigenwerte, die transformierte Matrix der unabhängigen Variablen, der transformierte Antwortvektor, die normierte Eigenwert-Matrix **M**, die Koordinaten der Extremstelle und die Koeffizienten der Regressionsgleichung sowie ihre Standardabweichungen abgerufen werden.

Tab. 9-1. Eigenschaften der Funktion *mdcanan*

mdcanan:	Kanonische Analyse zur Antwortflächenerforschung
Eingabe:	Matrix der unabhängigen Variablen, Antwortvektor
Ausgabe:	Kanonische Koeffizienten, kanonisch transformierte Matrix der unabhängigen Variablen und entsprechender Vektor der Antworten, Matrix der Eigenvektoren der Koeffizientenmatrix, Werte der unabhängigen Variablen an der Extremstelle, Funktionswert an der Extremstelle, Regressionskoeffizienten, Standardabweichung der Regressionskoeffizienten

9.2 Verwendung modellfremder Diagnoseparameter

Mittels Diagnoseparametern ist es möglich, zwischen konkurrierenden Modellen eine Auswahl zu treffen und teilweise auch Informationen für einen gezielten Modellaufbau zu erhalten. Dabei wird zwischen modelleigenen und modellfremden Diagnoseparametern unterschieden, wobei die modelleigenen bereits im Modell vorhanden sind, während die modellfremden extra eingeführt werden.

Auf die Entwicklung von Verfahren unter Verwendung modelleigener Diagnoseparameter ist hier verzichtet worden, da diese Methoden meistens auf sehr speziellen Umformungen der Modellgleichungen beruhen [9].
Im nachfolgenden werden zwei Methoden zur Unterscheidung zwischen zwei Einfachmodellen auf der Grundlage der Einführung modellfremder Diagnoseparameter vorgestellt.

9.2.1 Verfahren nach WILLIAMS und KLOOT

Verzeichnis: *mdiscrimination*
Funktionen: *mdextdiag.m*

Bei der Methode nach WILLIAMS und KLOOT [9] kann zwischen zwei Einfachmodellen unterschieden werden, indem ein modellfremder Diagnoseparameter als Teil eines linearen Tests eingeführt wird.
Dazu geht man von zwei zu den Modellen gehörenden Regressionsgleichungen

$$\begin{aligned} \hat{y}_1 &= f_1(\mathbf{b}_1, \mathbf{x}) \\ \hat{y}_2 &= f_2(\mathbf{b}_2, \mathbf{x}) \end{aligned} \tag{9-4}$$

aus, wobei die Parametervektoren beispielsweise über die Methoden aus Kap. 6 und Kap. 7 geschätzt werden können.
Der modellfremde Diagnoseparameter ergibt sich nun als Schätzwert für die Steigung λ einer Geradenfunktion z, für die gilt

$$z = y - \frac{1}{2} \cdot (\hat{y}_1 + \hat{y}_2) = \lambda \cdot (\hat{y}_1 - \hat{y}_2) \tag{9-5}$$

Der Diagnoseparameter kann dann folgendermaßen aus den beobachteten und vorhergesagten Werten geschätzt werden

$$\hat{\lambda} = \frac{\sum_{i=1}^{n} (\hat{y}_{1i} - \hat{y}_{2i}) \cdot [y_i - \frac{1}{2} \cdot (\hat{y}_{1i} + \hat{y}_{2i})]}{\sum_{i=1}^{n} (\hat{y}_{1i} - \hat{y}_{2i})^2} \tag{9-6}$$

und hat die Standardabweichung

$$s_\lambda = \frac{\sum_{i=1}^{n} y_i - \frac{1}{2} \cdot (\hat{y}_{1i} + \hat{y}_{2i}) - \hat{\lambda} \cdot (\hat{y}_{1i} - \hat{y}_{2i})}{(n-1) \cdot \sum_{i=1}^{n} (\hat{y}_{1i} - \hat{y}_{2i})^2} \tag{9-7}$$

Die dazugehörigen *t*-verteilten individuellen Vertrauensintervalle können über Gl. (6-7) bestimmt werden.
Die Anwendung dieses Verfahrens und die notwendigen Berechnungen erfolgen unter Vorgabe der beobachteten Werte und einer Konfidenzzahl für den Vertrauensbereich sowie unter der getrennten Eingabe der Antworten beider Modelle in der Funktion *mdextdiag* (vgl. Tab. 9-2).

Tab. 9-2. Eigenschaften der Funktion *mdextdiag*

mdextdiag:	Modellunterscheidung zwischen zwei Einfachmodellen über einen modellfremden Diagnoseparameter nach WILLIAMS und KLOOT
Eingabe:	Vektor der Beobachtungen, Antwortvektor Modell 1, Antwortvektor Modell 2, Konfidenzzahl*
Ausgabe:	Diagnoseparameter, Standardabweichung und Konfidenzintervall-halblängen für den Diagnoseparameter, Beurteilung zur Modellauswahl

* = Eingabe freigestellt

Über *mdextdiag* wird neben dem Wert des Diagnoseparameters und seiner Vertrauensintervalle auch die Beurteilung der Modelle direkt im MATLAB-Fenster ausgegeben. Die Ergebnisse lassen sich auf folgende Weise interpretieren [9]:

1. Wenn das Vertrauensintervall des Diagnoseparameters den Wert 0.5, aber nicht den Wert -0.5 einschließt, ist Modell 1 vorzuziehen.
2. Wenn das Vertrauensintervall des Diagnoseparameters den Wert -0.5, aber nicht den Wert 0.5 einschließt, ist Modell 2 vorzuziehen.
3. Liegen beide Werte im Vertrauensintervall, kann keine Aussage über die Wahl der Modelle getroffen werden.
4. Enthält das Vetrauensintervall weder 0.5 noch -0.5, sollte beiden Modellen in ihrer Anwendbarkeit mißtraut werden.

9.2.2 Likelihood-Quotienten-Test

Verzeichnis: *mdiscrimination*
Funktionen: *mdlhratio.m*

Der Likelihood-Quotienten-Test kann für die Unterscheidung zwischen zwei Modellen bei Einfachantworten angewendet werden und wird in der vorliegenden Programmsammlung über die Funktion *mdlhratio* ausgeführt [7,47]. Hierbei wird davon ausgegangen, daß die Parameter für beide Modelle bereits geschätzt worden und die Fehler normalverteilt sind.

In *mdlhratio* wird über das Likelihood-Verhältnis beider Modelle nach Vorgabe der beobachteten und vorhergesagten Werte eine Konfidenzzahl berechnet, durch die eine Modellunterscheidung möglich wird (vgl. Tab. 9-3).

Tab. 9-3. Eigenschaften der Funktion *mdlhratio*

mdlhratio:	Modellunterscheidung zwischen zwei Einfachmodellen über den Likelihood-Quotienten-Test
Eingabe:	Vektor der Beobachtungen, Antwortvektor beider Modelle
Ausgabe:	Konfidenzzahl

Bei diesem Testverfahren wird für beide Modelle eine Likelihood-Funktion gemäß Gl. (2-6) formuliert und der Likelihood-Quotient

$$\frac{L_1}{L_2} = \left(\frac{\sum_{i=1}^{n} (y_i - \hat{y}_{2i})^2}{\sum_{i=1}^{n} (y_i - \hat{y}_{1i})^2} \right)^{-\frac{n}{2}} \tag{9-8}$$

unter Berücksichtigung der Regressionsgleichungen nach Gl. (9-4) gebildet. Über diesen Quotienten kann eine Konfidenzzahl z als Maß für das Vertrauen in die Modelle gemäß

$$z = \frac{L_2}{L_1 + L_2} \tag{9-9}$$

ermittelt werden, wobei gilt:

$z \to 1$: Modell 2 ist zutreffend.
$z \to 0$: Modell 1 ist zutreffend.

Der Vorteil dieses Tests liegt darin, daß er ohne Kenntnis der Fehlervarianzen anzuwenden ist, weshalb er allerdings nicht ohne weiteres auf Modelle mit Mehrfachantworten ausgeweitet werden kann.

9.3 Einsatz von Versuchsplänen zur Modellunterscheidung

Bei der Unterscheidung zwischen mehreren Modellgleichungen kann die Lage der Versuchspunkte im Variablenraum einen größeren Einfluß haben als die Anzahl oder die Genauigkeit der Beobachtungswerte. Das bedeutet, die Auswahl der Modelle wird durch eine gezielte Durchführung von Versuchen erleichtert, bei denen sich die Modelle erheblich in ihren Antworten unterscheiden [9].

Für diese Form der Modelldiskriminierung wird auf Elemente der Versuchsplanung zurückgegriffen. Dadurch wird gestattet, mit möglichst wenig Versuchen unter den vorhandenen Modellen das statistisch gesehen beste auszuwählen.
Im folgenden werden Verfahren vorgestellt, die auf dieser Grundlage Modellunterscheidungen zwischen zwei und mehreren Modellen bei Einfach- und Mehrfachantwortsystemen ermöglichen. Ein Überblick über die Verfahren befindet sich in Tab. 9-4.

Tab. 9-4. Verfahren zur Modellunterscheidung unter Anwendung von Versuchsplänen

Verfahren nach	Einsatzgebiet	Voraussetzung
HUNTER und REINER (einfaches Kriterium)	Unterscheidung zwischen zwei Modellen bei Einfachantworten	Normalverteilung, unabhängige Antworten, konstante Varianz
HUNTER und REINER (strenges Kriterium)	Unterscheidung zwischen zwei Modellen bei Einfachantworten	Normalverteilung, unabhängige Antworten, konstante Varianz
BOX und HILL	Unterscheidung zwischen mehreren Modellen bei Einfachantworten	Normalverteilung, unabhängige Antworten, bekannte Fehlervarianz
HILL und HUNTER	Unterscheidung zwischen mehreren Modellen bei Mehrfachantworten	Normalverteilung, unabhängige Antworten, Varianz-Kovarianz-Matrix

Bei einer Durchführung von Experimenten zur Modellunterscheidung sind die zulässigen Werte der Einstellvariablen häufig experimentell oder physikalisch bedingt beschränkt. Dieses erfordert bei der Berechnung geeigneter Versuchspunkte den Einsatz von Optimierverfahren, deren Suchbereich eingegrenzt werden kann. Um wegen der notwendigen Beschränkung des Variablenbereichs nicht auf die Optimierfunktion *constr* aus der *Optimization Toolbox* von MATLAB angewiesen zu sein, werden die Optimierungen mit *fmins* durchgeführt, indem die Möglichkeit geschaffen wurde, den zulässigen Variablenbereich extern vorzugeben (vgl. Kap. 3.2 und 3.5).
Eine Änderung des voreingestellten Optimierverfahrens kann durch einen relativ einfachen Eingriff in das entsprechende Hauptprogramm vorgenommen werden, da auch hier die Funktion *optrout* eingesetzt wird und folglich das Optimierverfahren nur einmal programmintern gesetzt wird (vgl. Kap. 3.5).

9.3.1. Verfahren nach HUNTER und REINER

Verzeichnis: *mdiscrimination*
Funktionen: *mdhunrei.m*
mdhunreistr.m

Das Verfahren zur Modellauswahl nach HUNTER und REINER [9,49] kann zur Unterscheidung zwischen zwei Modellen bei Einfachantwortsystemen eingesetzt werden. Voraussetzung ist dabei die Unabhängigkeit und Normalverteilung der Beobachtungswerte sowie ihre konstante Varianz.
Zur Unterscheidung zwischen den Modellen nach Gl. (9-4) wird das Verhältnis der entsprechenden Likelihood-Funktionen aufgestellt. Dieses Verhältnis hängt maßgeblich von der Anzahl der bereits durchgeführten Versuche n ab. Das Kriterium von HUNTER und REINER bestimmt einen (n+1)-ten Versuchspunkt derartig, daß das Likelihood-Verhältnis maximal wird. Dabei reduziert sich das Kriterium auf die Berechnung des Restabweichungsquadrats

$$\Phi_1 = \sum_{i=1}^{n+1} \left(y_i - f_2\left(\mathbf{b}_2^{(n+1)}, \mathbf{x}_i\right)\right)^2 \rightarrow Max \qquad (9\text{-}10)$$

wobei in diesem Fall angenommen wird, daß Modell 1 korrekt ist. In dieses Kriterium geht die Bestimmung des Parametervektors $\mathbf{b}_2$ auf der Grundlage von (n+1) Versuchen ein, weshalb es zur Anwendung eines Iterationsalgorithmus kommt, der in Tab. 9-5 aufgeführt ist.

Tab. 9-5. Iterationsalgorithmus zum Verfahren nach HUNTER und REINER [9,49]

Annahme: Modell 1 ist korrekt!

1. Festlegung eines Startvektors für den (n+1)-ten Versuchspunkt $\mathbf{x}_{n+1}$
2. Berechnung der Modellantwort am (n+1)-ten Versuchspunkt für das Modell 1 nach

$$\hat{y}_{1,n+1} = f_1\left(b_1^{(n)}, \mathbf{x}_{n+1}\right)$$

3. Bestimmung der Parameterschätzwerte $b_2^{(n+1)}$ für Modell 2
4. Ermittlung eines optimalen $\mathbf{x}_{n+1}$ für ein maximales Φ_1 (Gl. (9.10))
5. Rückkehr zu Punkt 2, es sei denn, es wird kein $\mathbf{x}_{n+1}$ mehr berechnet, für das Φ_1 größer wird als der vorhergehende Φ_1-Wert (Abbruchbedingung).
6. Ausgabe von $\mathbf{x}_{n+1}$ als neuen Versuchspunkt

Das hier vorgestellte Verfahren beinhaltet mit Gl. (9-10) das Kriterium nach HUNTER und REINER in seiner strengen Form und ist in der Funktion

mdhunreistr umgesetzt (vgl. Tab. 9-6). Für die Parameterschätzung wird dabei die Funktion *multires* verwendet (vgl. Kap. 8.2). Sowohl für die Parameterschätzung als auch für die Optimierung der Versuchsstelle wird *fmins* eingesetzt. Das Schätzkriterium nach Gl. (9-10) befindet sich dabei in der Routine *mdhroptstr*, die vom Optimierverfahren *fmins* aufgerufen wird.

Tab. 9-6. Eigenschaften der Funktionen zur Modellunterscheidung nach HUNTER und REINER

mdhunreistr:	Berechnung eines Versuchspunkts zur Unterscheidung zwischen zwei Modellen bei Einfachantworten nach dem strengen Kriterium von HUNTER und REINER
Eingabe:	Matrix der unabhängigen Variablen, Antwortvektor, Modellgleichung 1, Parameterwerte zu 1, Modellgleichung 2, Parameterwerte zu 2, Schätzwert für den Versuchspunkt, Optimieroptionen*, zulässiger Parameterbereich*
Ausgabe:	Versuchspunkt zur Modellunterscheidung
mdhunrei:	Berechnung eines Versuchspunkts zur Unterscheidung zwischen zwei Modellen bei Einfachantworten nach dem vereinfachten Kriterium von HUNTER und REINER
Eingabe:	Modellgleichung 1, Parameterwerte zu 1, Modellgleichung 2, Parameterwerte zu 2, Schätzwert für den Versuchspunkt, Optimieroptionen*, zulässiger Parameterbereich*
Ausgabe:	Versuchspunkt zur Modellunterscheidung

* = Eingabe freigestellt

Über die Routine *mdhunrei* kann ein modifiziertes Kriterium nach HUNTER und REINER eingesetzt werden. Dazu wird die Schätzfunktion nach Gl. (9-10) vereinfacht, indem in die Berechnungen nicht der Parameterschätzvektor $\mathbf{b}_2$ auf der Grundlage von (n+1), sondern nur von den bereits durchgeführten n Versuchen einfließt. Damit vereinfacht sich Gl. (9-10) zu

$$\Phi = \left(f_1\left(\mathbf{b}_1^{(n)}, \mathbf{x}_{n+1}\right) - f_2\left(\mathbf{b}_2^{(n)}, \mathbf{x}_{n+1}\right)\right)^2 \rightarrow Max \qquad (9\text{-}11)$$

Dieses modifizierte Kriterium verringert den Rechenaufwand gegenüber Gl. (9-10) durch die fehlende Iteration und ist unabhängig davon, welches Modell als richtig angesehen wird. Die Funktion *mdhunrei* entspricht in ihrer Struktur und Eigenschaften ansonsten *mdhunreistr* (vgl. Tab. 9-6).

Beide Kriterien können grundsätzlich auch für ein sequentielles Vorgehen während der Versuchsplanung eingesetzt werden, wobei der Experimentator entscheiden muß, ob ein betrachtetes Modell als korrekt angesehen wird oder weitere Versuche durchgeführt werden müssen.

9.3.2 Verfahren nach BOX und HILL

Verzeichnis: *mdiscrimination*
Funktionen: *mdboxhill.m*
mdbhapost.m

Die Methode nach BOX und HILL [9,50] zur Unterscheidung zwischen mehreren Modellen bei Einfachantworten stellt zwar eine Verallgemeinerung des Verfahrens nach HUNTER und REINER dar, aber es werden zusätzliche Informationen wie die Varianzen der Antworten benötigt (vgl. Kap. 9.3.1). Mit diesem Verfahren wird ein neuer Versuchspunkt so ausgewählt, daß der Abstand der Grenzen der Vertrauensbereiche der Modelle und nicht der Unterschied in den Modellwerten wie in Kap. 9.3.1 möglichst maximal ist.

Um das (n+1)-te Experiment zu planen, durch das eine optimale Unterscheidung zwischen q konkurrierenden Modellen möglich wird, müssen die Einstellvariablen $\mathbf{x}_{n+1}$ so gewählt werden, daß die Funktion

$$D_0 = \sum_{i=1}^{q-1} \sum_{j=i+1}^{q} p_i^{(n)} p_j^{(n)} \left(\frac{\left(\sigma_i^2 - \sigma_j^2\right)^2}{\left(\sigma^2 + \sigma_i^2\right) \cdot \left(\sigma^2 + \sigma_j^2\right)} + \left(\hat{y}_{i,n+1} - \hat{y}_{j,n+1}\right)^2 \left(\frac{1}{\sigma^2 + \sigma_i^2} + \frac{1}{\sigma^2 + \sigma_j^2} \right) \right) \quad (9\text{-}12)$$

einen maximalen Wert annimmt. Dabei handelt es sich bei $p_i^{(n)}$ um die a-priori-Wahrscheinlichkeit für die Richtigkeit des i-ten Modells vor Durchführung des (n+1)-ten Versuchs. Ferner ist σ^2 die Fehlervarianz und σ_i^2 die Varianz der Antwort an der Stelle $\mathbf{x}_{n+1}$ auf der Grundlage des i-ten Modells.
Um Gl. (9-12) anzuwenden, muß die Fehlervarianz oder ein Schätzwert der Fehlervarianz bekannt sein. Die Varianz der Schätzfunktion für $\hat{y}_{i,n+1}$ kann nach Gl. (9-13) über die Fehlervarianz und die Jacobi-Matrizen für das i-te Modell an den einzelnen Versuchspunkten bestimmt werden, wobei die Jacobi-Matrizen für die ersten n Versuchspunkte zur zweidimensionalen Matrix $\mathbf{J}$ zusammengefaßt werden, und die Jacobi-Matrix am (n+1)-ten Versuchspunkt mit $\mathbf{j}$ bezeichnet wird.

$$\sigma_i^2 \approx s_i^2 = \mathbf{j}_{n+1,i}^T \cdot \left(\mathbf{J}_i^T \cdot \mathbf{J}_i\right)^{-1} \cdot \mathbf{j}_{n+1,i} \cdot \sigma^2 \quad (9\text{-}13)$$

Die Bestimmung der Jacobi-Matrix geschieht unter Verwendung der Parameterschätzwerte für das entsprechende Modell. Die Anwendung von Gl. (9-13) ist jedoch nur zulässig, wenn die Modellfunktionen als angenähert linear in den Parametern betrachtet werden können.

Der Einsatz des Verfahrens von BOX und HILL wird durch zwei Funktionen, *mdboxhill* und *mdbhapost*, ermöglicht (vgl. Tab. 9-7). Hierbei wird über *mdboxhill* der neue Versuchspunkt über Gl. (9-12) und (9-13) bestimmt. Die Schätzkriterien sind in der Routine *mdbhopt* abgelegt. Die Berechnungen der Jacobi-Matrizen erfolgen über *jacsym* und *jacsymval* (vgl. Kap 4.2).

Tab. 9-7. Eigenschaften der Funktionen zur Modellunterscheidung nach BOX und HILL

mdboxhill:	Berechnung eines Versuchspunkts zur Unterscheidung verschiedenen Einfachantwortmodellen nach dem Kriterium von BOX und HILL
Eingabe:	Matrix der unabhängigen Variablen, Antwortmatrix, Modellgleichungen, *cell array* mit den Parametervektoren zu den Modellen, Schätzwert für den neuen Versuchspunkt, a-priori-Wahrscheinlichkeit der Modelle*, Schätzwert der Fehlervarianz*, Optimieroptionen*, zulässiger Schätzbereich*
Ausgabe:	Versuchspunkt zur Modellunterscheidung, a-priori-Wahrscheinlichkeiten der Modelle
mdbhapost:	Bestimmung der a-posteriori-Wahrscheinlichkeit der verschiedenen Modelle bei Anwendung von *mdboxhill*
Eingabe:	Matrix der unabhängigen Variablen, Antwortmatrix, Modellgleichungen, *cell array* mit den Parametervektoren zu den Modellen, a-priori-Wahrscheinlichkeit der Modelle *, Schätzwert der Fehlervarianz*
Ausgabe:	a-posteriori-Wahrscheinlichkeiten der Modelle

* = Eingabe freigestellt

Sind die a-priori-Wahrscheinlichkeiten der Modelle als Voraussetzung für *mdboxhill* nicht durch Vorversuche bekannt, werden die Modelle programmintern als gleich wahrscheinlich betrachtet.
Eine Modellunterscheidung wird möglich, wenn die a-posteriori-Wahrscheinlichkeit eines der Modelle gegen eins strebt. Nach dem durchgeführten Experiment können die a-posteriori-Wahrscheinlichkeiten $p_i^{(n+1)}$ mit *mdbhapost* ermittelt werden. Dies geschieht über das Bayessche Theorem

$$p_i^{(n+1)} = \frac{p_i^{(n)} \cdot p_i}{\sum_{i=1}^{q} p_i^{(n)} \cdot p_i} \tag{9-14}$$

wobei p_i die Wahrscheinlichkeitsdichtefunktion von y_{n+1} nach Gl. (9-15) ist.

$$p_i = \frac{1}{\sqrt{2\pi \cdot \left(\sigma^2 - \sigma_i^{\,2}\right)}} \cdot \exp\left(- \frac{\left(y_{n+1} - \hat{y}_{i,n+1}\right)^2}{2 \cdot \left(\sigma^2 - \sigma_i^{\,2}\right)} \right) \tag{9-15}$$

Erweist sich kein Modell als sehr wahrscheinlich, muß mit *mdboxhill* ein weiterer Versuch geplant werden, wobei die berechneten a-posteriori-Wahrscheinlichkeiten als a-priori-Wahrscheinlichkeiten für die neue Versuchsplanung zur Verfügung stehen.
Dieses sequentielle Vorgehen aus Versuchsplanung, -durchführung und –auswertung wird so lange wiederholt, bis eine eindeutige Unterscheidung möglich ist.

9.3.3 Verfahren nach HILL und HUNTER

Verzeichnis: *mdiscrimination*
Funktionen: *mdhillhun.m*
mdhhapost.m

Zur Unterscheidung zwischen mehreren Modellen bei Mehrfachantworten kann ein Algorithmus nach HILL und HUNTER angewendet werden [9]. Dieses Verfahren entspricht in seinem Vorgehen der Methode von BOX und HILL und kann als dessen Verallgemeinerung betrachtet werden. Da bei Mehrfachantwortsystemen die Ansprüche an die Vorgaben steigen und der interne Rechen- und Programmieraufwand zunimmt, ist es zweckmäßig, neben diesem allgemeinen Verfahren auch die bisher vorgestellten spezielleren Methoden zur Modellunterscheidung in diese Programmsammlung aufzunehmen.

Für die Anwendung des Verfahrens nach HILL und HUNTER stehen die Funktionen *mdhillhun* und *mdhhapost* zur Verfügung (vgl. Tab. 9-8). Die Programmstrukturen und die sequentielle Anwendung dieser Funktionen stimmen weitgehend mit den Funktionen *mdboxhill* und *mdbhapost* überein (vgl. Kap. 9.3.2).

Tab. 9-8. Eigenschaften der Funktionen zur Modellunterscheidung nach HILL und HUNTER

mdhillhun:	Berechnung eines Versuchspunkts zur Unterscheidung verschiedener Mehrfachantwortmodelle nach dem Kriterium von HILL und HUNTER
Eingabe:	Matrix der unabhängigen Variablen, Antwortmatrix, Modellgleichungen, *cell array* mit den Parametervektoren zu den Modellen, Schätzwert für den neuen Versuchspunkt, a-priori-Wahrscheinlichkeit der Modelle*, Schätzwert der Varianz-Kovarianz-Matrix*, Optimieroptionen*, zulässiger Schätzbereich*
Ausgabe:	Versuchspunkt zur Modellunterscheidung, a-priori-Wahrscheinlichkeiten der Modelle
mdhhapost:	Bestimmung der a-posteriori-Wahrscheinlichkeit der verschiedenen Modelle bei Anwendung von *mdhillhun*
Eingabe:	Matrix der unabhängigen Variablen, Antwortmatrix, Modellgleichungen, *cell array* mit den Parametervektoren zu den Modellen, a-priori-Wahrscheinlichkeit der Modelle *, Schätzwert der Varianz-Kovarianz-Matrix der Meßfehler zwischen den Antworten*
Ausgabe:	a-posteriori-Wahrscheinlichkeiten der Modelle

* = Eingabe freigestellt

Die Berechnungsgrundlage für die Funktionen *mdhillhun* und *mdhhapost* stellen Anpassungen der Kriterien aus Kap. 9.3.2 auf Mehrfachantwortsysteme dar.
Die Schätzfunktion für den neuen Versuchspunkt lautet bei einem System von q Modellen und jeweils r Antworten in diesem Fall

$$D_\mu = \frac{1}{2} \cdot \sum_{i=1}^{q-1} \sum_{j=1}^{q} p_i^{(n)} p_j^{(n)} \begin{pmatrix} tr\left(\overline{\mathbf{V}}_i \cdot \overline{\mathbf{V}}_j^{-1} + \overline{\mathbf{V}}_j \cdot \overline{\mathbf{V}}_i^{-1} - 2 \cdot \mathbf{I}_r\right) + \\ + \left(\hat{\mathbf{y}}_{i,n+1} - \hat{\mathbf{y}}_{j,n+1}\right)^T \cdot \left(\overline{\mathbf{V}}_j^{-1} + \overline{\mathbf{V}}_i^{-1}\right) \cdot \left(\hat{\mathbf{y}}_{i,n+1} - \hat{\mathbf{y}}_{j,n+1}\right) \end{pmatrix} \tag{9-16}$$

Dabei ergeben sich die Matrizen $\overline{\mathbf{V}}_i$ als Summe aus der Varianz-Kovarianz-Matrix der Meßfehler zwischen den Antworten $\mathbf{V}$ und der Varianz-Kovarianz-Matrix $\mathbf{V}_i$ der Schätzfunktion für $\hat{y}_{i,n+1}$, wobei analog zu Gl. (9-13) gilt

$$\mathbf{V}_i = \mathbf{J}_{n+1,i} \cdot \left(\sum_{u=1}^{r} \sum_{v=1}^{r} V_{uv}^{-1} \cdot \mathbf{J}_{iu}^T \cdot \mathbf{J}_{iv} \right)^{-1} \cdot \mathbf{J}_{n+1,i}^T \tag{9-17}$$

In Gl. (9-17) ist $\mathbf{J}_{i,j}$ die Jacobi-Matrix der j-ten Antwort im i-ten Modell für alle Versuchspunkte, während $\mathbf{J}_{n+1,i}$ die Jacobi-Matrix für den (n+1)-ten Versuch und für die r Modellgleichungen des i-ten Modells ist.

Die a-posteriori-Wahrscheinlichkeiten werden in *mdhhapost* nach Gl. (9-14) berechnet. Allerdings ergibt sich die Wahrscheinlichkeitsdichtefunktion hierbei für $\hat{\mathbf{y}}_{i,n+1}$ zu

$$p_i = \frac{\left(\det\left(\overline{\mathbf{V}}_i\right)\right)^{-\frac{1}{2}}}{\sqrt{(2\pi)^r}} \cdot \exp\left(-\frac{1}{2} \cdot \left(\mathbf{y}_{n+1} - \hat{\mathbf{y}}_{i,n+1}\right)^T \cdot \overline{\mathbf{V}}_i^{-1} \cdot \left(\mathbf{y}_{n+1} - \hat{\mathbf{y}}_{i,n+1}\right) \right) \tag{9-18}$$

Auch bei diesem Verfahren erweist sich ein Modell als besonders wahrscheinlich, wenn seine a-posteriori-Wahrscheinlichkeit im Laufe der Versuche gegen eins strebt.

9.3.4 Anwendungsbeispiel

Aufgrund der allgemeinen Bedeutung des Verfahrens nach HILL und HUNTER soll die Einsatzfähigkeit der Funktion *mdhillhun* näher untersucht werden. Hierfür wird ein in [9] beschriebenes hypothetisches Beispiel nach HUNTER und WICHERN betrachtet. Dabei handelt es sich um eine heterogen katalysierte Gasphasenreaktion, die sich schematisch folgendermaßen formulieren läßt

$$\begin{aligned} \mathrm{A} + \mathrm{B} &\xrightarrow{k_1} \mathrm{C} + \mathrm{D} \\ \mathrm{A} + \mathrm{B} &\xrightarrow{k_2} \mathrm{C} + \mathrm{E} \end{aligned} \tag{9-19}$$

Es ist angenommen worden, daß die Reaktionsgeschwindigkeiten direkt zugänglich sind, irreversible Reaktionen vorliegen und die Produkte nicht adsorbiert werden.

Im folgenden sind die möglichen Modelle nach [9] in ybx-Schreibweise formuliert:

Modell 1:

$$y(1) = \frac{b(1) \cdot b(3) \cdot b(4) \cdot x(1) \cdot x(2)}{\left(1 + b(3) \cdot x(1) + b(4) \cdot x(2)\right)^2} \qquad y(2) = \frac{b(2) \cdot b(3) \cdot b(4) \cdot x(1) \cdot x(2)}{\left(1 + b(3) \cdot x(1) + b(4) \cdot x(2)\right)^2} \tag{9-20}$$

Modell 2:

$$y(1) = \frac{b(1) \cdot b(3) \cdot b(4) \cdot x(1) \cdot x(2)}{\left(1 + b(3) \cdot x(1) + b(4) \cdot x(2)\right)^2} \qquad y(2) = \frac{b(2) \cdot b(3) \cdot x(1) \cdot x(2)}{1 + b(3) \cdot x(1) + b(4) \cdot x(2)} \tag{9-21}$$

Modell 3:

$$y(1) = \frac{b(1) \cdot b(3) \cdot x(1) \cdot x(2)}{1 + b(3) \cdot x(1)} \qquad y(2) = \frac{b(2) \cdot b(3) \cdot b(4) \cdot x(1) \cdot x(2)}{\left(1 + b(3) \cdot x(1) + b(4) \cdot x(2)\right)^2} \qquad (9\text{-}22)$$

Hierbei stehen $b(1)$ und $b(2)$ für die Geschwindigkeitskonstanten k_1 und k_2 und $b(3)$ und $b(4)$ für die Adsorptionsgleichgewichtskonstanten. Die Partialdrücke der Komponenten A und B werden durch die unabhängigen Variablen $x(1)$ und $x(2)$ ausgedrückt.

In der Simulation des Experiments ist Modell 3 als korrekt mit den Parametern

$$b(1) = 0.0005 \qquad b(2) = 0.16 \qquad b(3) = 15 \qquad (9\text{-}23)$$

angenommen worden. Die Beobachtungswerte sind auf dieser Grundlage als unabhängig und normalverteilt mit den Standardabweichungen

$$\sigma_1 = 3.162\,e{-}6 \qquad \sigma_2 = 1.0\,e{-}4 \qquad (9\text{-}24)$$

berechnet worden.
Für die Startversuche hat man einen 2^2-Faktorplan aufgestellt. Die dafür simulierten Daten befinden sich in Tab. 9-9.

Tab. 9-9. Simulierte Daten für ein Modellunterscheidungsproblem [9]

Versuch	Einstellvariablen		Antworten	
	$x(1)$	$x(2)$	$y(1)$	$y(2)$
1	0.025	0.025	4.639e-6	0.846e-3
2	0.075	0.025	8.797e-6	1.062e-3
3	0.025	0.075	7.135e-6	2.364e-3
4	0.075	0.075	2.3587e-5	3.049e-3

Die zur Auswertung notwendige Parameterschätzung ist mit der Funktion *multires* unter Verwendung von *fmins* als Optimierverfahren in Standardeinstellungen durchgeführt worden (vgl. Kap. 8.2). Als Schätzkriterium ist das Determinantenkriterium verwendet worden (vgl. Tab. 8-1).
Zur Modellunterscheidung hat man die Funktion *mdhillhun* eingesetzt. Für die Vorversuche ist jedes Modell als gleich wahrscheinlich betrachtet worden. Die zulässigen Partialdrücke sind auf 0.075 begrenzt worden.

Die Berechnung der a-posteriori-Wahrscheinlichkeiten erfolgte über die Funktion *mdhhapost*.

Im Rahmen der Auswertung sind zunächst die Parameter aus den Vorversuchen geschätzt, ein neuer Versuchspunkt berechnet und die a-posteriori-Wahrscheinlichkeiten ermittelt worden. Um ein sequentielles Vorgehen in der Modellunterscheidung zu simulieren, sind aus dem neuen Versuchspunkt über Modell 3 neue Beobachtungswerte generiert worden. Mit diesen zusätzlichen Beobachtungswerten sind neue Modellparameter, ein weiterer optimaler Versuchspunkt und die a-posteriori-Wahrscheinlichkeiten ermittelt worden. Dieser Ablauf ist sechsmal wiederholt worden.

Die Ergebnisse bezüglich der a-posteriori-Wahrscheinlichkeiten für die verschiedenen Modelle sind in Tab. 9-10 aufgeführt und in Abb. 9-1 graphisch dargestellt.

Tab. 9-10. A-posteriori-Wahrscheinlichkeiten bei Anwendung von *mdhillhun*

Versuch	$p_1^{(n)}$	$p_2^{(n)}$	$p_3^{(n)}$
0	0.33333	0.33333	0.33333
1-4	0.38967	0.11369	0.49664
5	0.80881	0.07640	0.11480
6	0.06760	0.00157	0.93083
7	0.01780	0.00068	0.98152
8	0.00001	0.00023	0.99976
9	0.00001	0.00012	0.99987

Die erhaltenen Ergebnisse zeigen erwartungsgemäß die Überlegenheit von Modell 3 gegenüber den anderen Modellen.

Allerdings wird auch deutlich, daß lediglich Tendenzen in den a-posteriori-Wahrscheinlichkeiten zur Modellunterscheidung nicht ausreichen. Vielmehr muß die Wahrscheinlichkeit angenähert einen Wert von eins erreichen. In diesem Beispiel hätte ein Abbruch des Verfahrens nach dem fünften Versuchspunkt gegebenenfalls zur Bestätigung des falschen Modell 1 geführt.

Es genügt somit in diesem Fall die Durchführung von drei bis fünf weiteren Versuchen, um die Modellfrage im Rahmen der betrachteten Modelle eindeutig zu klären.

Einen nicht unerheblichen Einfluß hat hierbei die weitgehend experimentell bedingte Beschränkung des zulässigen Bereichs der Einstellvariablen. Je größer

dieser Bereich gewählt wird, desto mehr Versuche sind notwendig. In einem vergleichbaren Simulationsexperiment, in dem der Druckbereich auf die Einheit 1 erweitert worden ist, mußten zehn zusätzliche Versuche berechnet werden, um zumindest eine eindeutige Tendenz in der Modellauswahl zu erkennen.

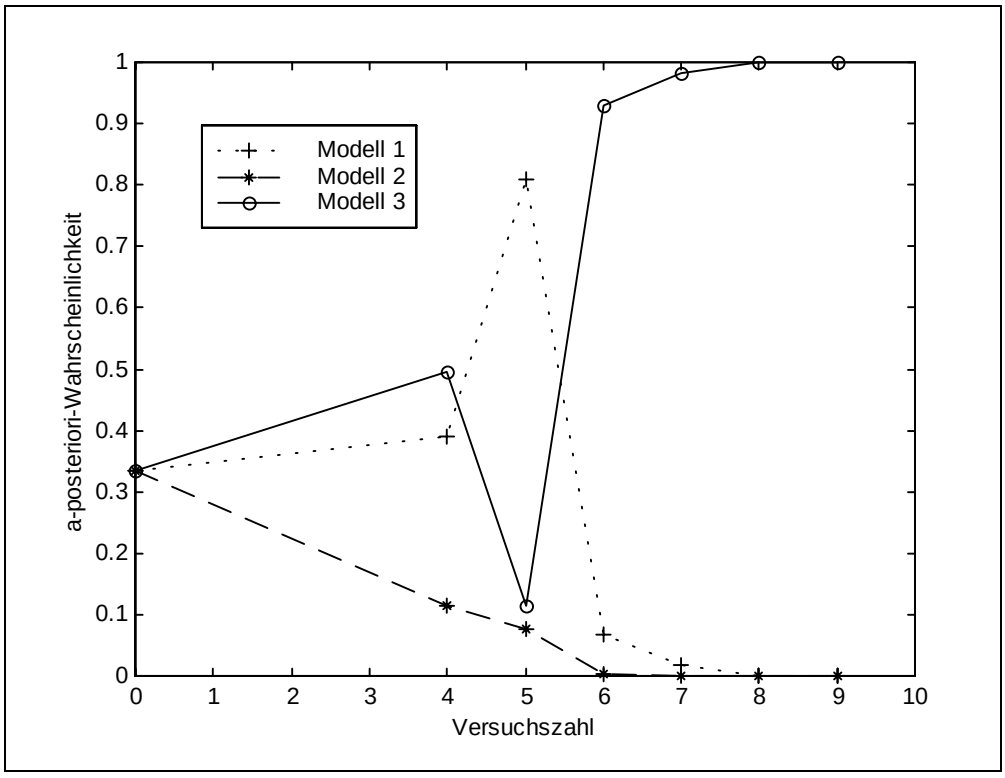

Abb. 9-1. Darstellung der a-posteriori-Wahrscheinlichkeiten in Abhängigkeit der Versuchszahl in MATLAB

Das vorliegende Modellunterscheidungsproblem kann grundsätzlich auch als Einfachantwortsystem behandelt und mit *mdboxhill* bearbeitet werden (vgl. Kap. 9.3.2). Dieses hat zwar den Vorteil, daß ein einfacheres Kriterium verwendet wird, allerdings gehen dabei Informationen verloren, wodurch die Anzahl der zusätzlichen Versuche und folglich der experimentelle Aufwand zunimmt [9].

10 Verfahren zur Versuchsplanung

Unter dem Begriff der Versuchsplanung werden Methoden und Verfahren zusammengefaßt, die Versuchspunkte so festlegen, daß bei einem zu untersuchenden Prozeß der experimentelle und rechnerische Aufwand möglichst gering und der informative Nutzen relativ hoch ist [8,30,51]. Die Aufgabe der Versuchsplanung besteht dabei hauptsächlich in

- der Feststellung signifikanter Einflußgrößen,
- der Parameterpräzisierung,
- der Modellunterscheidung,
- der Antwortflächenerforschung und
- der Prozeßoptimierung.

Da der fachliche Bezug dieser Arbeit in der Reaktionskinetik liegt, werden hier nur die ersten vier Einsatzgebiete der Versuchsplanung berücksichtigt, wobei die Verfahren zur Antwortflächenerforschung und Modellunterscheidung bereits in Kap. 9 behandelt worden sind. Aus diesen Gründen werden im folgenden vor allem Verfahren zur Faktorplanauswertung sowie zur Parameterpräzisierung vorgestellt.
Die Behandlung der Faktorplanauswertung erfolgt wegen der Vollständigkeit dieser Programmsammlung und soweit es für die Reaktionskinetik relevant ist. Bezüglich der Aufstellung von Faktorplänen und einer weitergehenden Betrachtung spezieller Faktorpläne wird auf die *Statistics Toolbox* von MATLAB verwiesen [17].
Die Verfahren zur Parameterpräzisierung umfassen sowohl simultane als auch sequentielle Versuchsplanungstechniken.

10.1 Versuchspläne

In den folgenden Betrachtungen werden wegen ihrer häufigen Verbreitung ausschließlich Versuchspläne behandelt, deren Antworten polynomial von den Einstellvariablen abhängen [30,31]. Die allgemeine Form des zugrunde liegenden Modells lautet hierbei

$$\begin{aligned}\eta = \beta_0 \cdot x_0 + \beta_1 \cdot x_1 + \ldots + \beta_m \cdot x_m + \beta_{11} \cdot x_1^{\,2} + \ldots + \beta_{mm} \cdot x_m^{\,2} + \\ + \beta_{12} \cdot x_1 \cdot x_2 + \ldots + \beta_{m-1,m} \cdot x_{m-1} \cdot x_m + \ldots + \beta_{111} \cdot x_1^{\,3} + \ldots\end{aligned} \tag{10-1}$$

In Gl. (10-1) stellt x_0 eine Scheinvariable dar, die im allgemeinen den Wert eins annimmt.

Bei der Behandlung von Versuchsplänen werden die funktionell unabhängigen Variablen als Faktoren und die funktionell abhängigen Variablen als Wechselwirkungen bezeichnet. Zu den Wechselwirkungen gehören unter anderem die gemischten und quadratischen Terme der Einstellvariablen in Gl. (10-1). Die Wirkung eines Faktors oder einer Wechselwirkung auf eine Antwort stellt einen sogenannten Effekt dar. Ein Maß für einen solchen Effekt ist die Größe des entsprechenden Koeffizienten in Gl. (10-1).
Die verschiedenen Einstellwerte der Faktoren in einem Versuchsplan werden als Niveau aufgefaßt. Im allgemeinen charakterisiert man die Niveaus mit ganzen Zahlen und ordnet der niedrigsten Einstellung den Wert Null zu.
Ein Versuchsplan, der aus *m* Faktoren aufgebaut ist und über den Koeffizienten bis zum Grad *d* bestimmt werden können, ist ein *m*-dimensionaler Faktorplan des Grades *d*. Die Zahl der dafür notwendigen Niveaus beträgt mindestens (*d*+1).
Zur Bearbeitung von Versuchsplänen werden die funktionell unabhängigen Variablen in der Planmatrix **X** zusammengefaßt, wobei die Variablen in den Spalten und ihre Einstellwerte an den verschiedenen Versuchspunkten in den Zeilen aufgeführt sind.
Zur Verringerung von Rechenaufwand und Rundungsfehlern werden die unabhängigen Variablen zentriert und normiert. Dieser Vorgang läßt sich für die Komponenten der Planmatrix X_{ij} im allgemeinen mathematisch folgendermaßen beschreiben

$$\tilde{X}_{ij} = \frac{X_{ij} - \frac{1}{2} \cdot \left(\max_i \left(X_{ij} \right) + \min_i \left(X_{ij} \right) \right)}{\frac{1}{2} \cdot \left(\max_i \left(X_{ij} \right) - \min_i \left(X_{ij} \right) \right)} \tag{10-2}$$

Durch eine solche Zentrierung und Normierung werden die Niveaus auf das Intervall zwischen +1 und –1 beschränkt. [8,48,52]
Für die Auswertung eines Faktorplans muß eine sogenannte Berechnungsmatrix aufgestellt werden. Die Spalten dieser Matrix sind den verschiedenen Faktoren und Wechselwirkungen sowie der Scheinvariable zugeordnet. Die Zeilen repräsentieren die unterschiedlichen Versuchsniveaus. Unter Berücksichtigung der Berechnungsmatrix $\mathbf{X}_C$, des Parametervektors $\boldsymbol{\beta}$ und des Antwortvektors $\boldsymbol{\eta}$ läßt sich Gl. (10-1) für *n* Versuchspunkte auf folgende Weise in Vektorschreibweise formulieren

$$\boldsymbol{\eta} = \boldsymbol{\beta} \cdot \mathbf{X}_C \tag{10-3}$$

Ein Versuchsplan sollte insgesamt die Forderungen nach Orthogonalität, Rotierbarkeit und der Möglichkeit zum Modellausbau erfüllen. Durch diese Anforderungen wird ein hoher Informationsgehalt des Versuchsplans gewährleistet und signifikante Einflußgrößen können ohne größeren Rechenaufwand bestimmt werden, da unter diesen Voraussetzungen unter anderem die Koeffizienten in Gl. (10-1) unkorreliert sind.
In den nachfolgenden Abschnitten werden Funktionen zur Aufstellung der Berechnungsmatrix, zur Analyse von Versuchsplänen und zur Auswertung von faktoriellen Versuchsplänen vorgestellt.

10.1.1 Aufstellung der Berechnungsmatrix

Verzeichnis: *exdesign*
Funktionen: *edcalcm.m*
edcalcmo.m

Für die Aufstellung der Berechnungsmatrix können die Funktionen *edcalcm* und *edcalcmo* eingesetzt werden.
Bei *edcalcm* wird die Planmatrix unter Angabe des Grades der reinen und gemischten Terme eingelesen (vgl. Gl. (10-1)). Intern wird die Berechnungsmatrix aufgestellt. Die Anordnung der Faktoren und Wechselwirkungen erfolgt gemäß Gl. (10-1), d. h. in der ersten Spalte befindet sich die Scheinvariable, darauf folgen die Faktoren, anschließend die Wechselwirkungen. Allgemein gilt dabei, daß die gemischten Terme mit einem Gesamtgrad d auf die reinen Terme vom Grad d folgen. Die getrennte Vorgabe des Grades bei den gemischten und reinen Termen ermöglicht eine Freizügigkeit in der Modellvorgabe, da nach Bedarf gemischte oder reine Terme unabhängig voneinander berücksichtigt werden können.
Die Funktion *edcalcm* läßt sich vor allem zur Aufstellung der Berechnungsmatrix auf die in der Praxis besonders häufig auftretenden Faktorpläne mit zwei Niveaus anwenden [51]. Die wichtigsten Eigenschaften von *edcalcm* sind in Tab. 10-1 aufgeführt.

Bei der Auswertung von Faktorplänen sollte die Berechnungsmatrix orthogonal sein. Hierdurch können die Parameter unabhängig voneinander geschätzt werden, was die Identifikation von Einflußgrößen erleichtert. Diese Orthogonalitätsforderung ist bei Anwendung von *edcalcm* auf Faktorpläne höheren Grades jedoch nicht mehr unbedingt erfüllt [8]. Unter Umständen kann dieses Problem gelöst werden, indem die Einstellvariablen nicht direkt bei der

Berechnung der Spalten der Berechnungsmatrix berücksichtigt werden, sondern diesbezüglich über einen Satz orthogonaler Polynome eingehen [8]. Dabei kann nicht jeder Satz an orthogonalen Polynomen verwendet werden, sondern nur solche, bei denen die Rücktransformation nach der Parameterschätzung problemlos möglich ist. Mit der Funktion *edcalcmo* wird durch Zugriff auf die *Extended Symbolic Toolbox* eine Berechnungsmatrix auf der Basis der Legendre-Polynome erzeugt [19]. Neben der Berechnungsmatrix gibt diese Funktion die Berechnungsgleichungen für die Spalten in Form eines c*ell array* aus. (Vgl. Tab. 10-1)

Tab. 10-1. Eigenschaften der Funktionen *edcalcm* und *edcalcmo*

edcalcm:	Aufstellung der Berechnungsmatrix für einen Faktorplan
Eingabe:	Planmatrix, Grad der reinen Terme, Grad der gemischten Terme
Ausgabe:	Berechnungsmatrix
edcalcmo:	Aufstellung einer Berechnungsmatrix für einen Faktorplan unter Verwendung von Legendre-Polynomen
Eingabe:	Planmatrix, Grad der reinen Terme, Grad der gemischten Terme
Ausgabe:	Berechnungsmatrix, Berechnungsgleichungen für die einzelnen Spalten

Bei der programmtechnischen Umsetzung der vorgestellten Funktionen hat die Hauptschwierigkeit in der Entwicklung eines Programmcodes bestanden, der es ermöglicht, sich die mathematischen Ausdrücke zur Spaltenberechnung für einen beliebigen Grad des Modells zu erzeugen und die Berechnungsmatrix entsprechend zu sortieren. Dabei hat sich die Stringverarbeitung von MATLAB als besonders vorteilhaft erwiesen, da so relativ einfach ein sich selbst generierender Programmcode entwickelt worden ist, der den Anforderungen gerecht wird.

10.1.2 Analyse von Versuchsplänen

Verzeichnis: *exdesign*
Funktionen: *edanal.m*

Über die Funktion *edanal* können unter obigen Aspekten aufgestellte Versuchspläne analysiert werden. Hierfür müssen die Planmatrix sowie die Grade der reinen und gemischten Glieder nach Gl. (10-1) vorgegeben werden. Als Ergebnis erhält man dafür die Berechnungsmatrix, die Weiten der unabhängigen Variablen und die Größe des Versuchsplans sowie eine Aussage über die Homogenität des Versuchsplans. (Vgl. Tab. 10-2)

Tab. 10-2. Eigenschaften der Funktion *edanal*

edanal:	Analyse eines gegebenen Versuchsplans
Eingabe:	Planmatrix, Grad der reinen Glieder, Grad der gemischten Glieder
Ausgabe:	Berechnungsmatrix, Weiten der transformierten und untransformierten Variablen, Größe des Versuchsplans

In *edanal* wird die vorgegebene Planmatrix transformiert und über die Funktion *edcalcm* die Berechnungsmatrix aufgestellt (vgl. Kap. 10.1.1).
Zur Charakterisierung der Lage der Versuchspunkte im Variablenraum können die Weiten der untransformierten und transformierten Variablen w_j und $\tilde{w}_j$ für n Versuchspunkte unter Beachtung von Gl. (10-2) berechnet werden nach

$$w_j^{\,2} = \frac{1}{n} \cdot \sum_{i=1}^{n} \left(X_{ij} - \frac{1}{2} \cdot \left(\max_i \left(X_{ij} \right) + \min_i \left(X_{ij} \right) \right) \right)^2 \tag{10-4}$$

$$\tilde{w}_j^{\,2} = \frac{1}{n} \cdot \sum_{i=1}^{n} \tilde{X}_{ij}^{\,2} \tag{10-5}$$

Über Gl. (10-5) kann die Größe $\tilde{w}$ des Versuchsplans für m Parameter folgendermaßen bestimmt werden

$$\tilde{w}^2 = \frac{1}{m} \cdot \sum_{j=1}^{m} \tilde{w}_j^{\,2} \tag{10-5}$$

Ein Versuchsplan gilt nun als homogen, wenn alle Weiten der transformierten Variablen der Größe des Versuchsplans entsprechen. Dies wird in *edanal* überprüft und das Ergebnis im MATLAB-Fenster ausgegeben.

10.1.3 Auswertung von faktoriellen Versuchsplänen

Verzeichnis: *exdesign*
Funktionen: *edfulfac2n.m*
edfulfacn.m

Faktorielle Versuchspläne werden eingesetzt, wenn man über das zu untersuchende System relativ wenig weiß oder signifikante Einflußgrößen feststellen möchte. Dabei unterscheidet man zwischen vollständigen Faktorplänen und Teilfaktorplänen [8]. Bei den vollständigen Faktorplänen werden alle Kombinationsmöglichkeiten der Niveaus der Faktoren berücksichtigt, während in Teilfaktorplänen gewöhnlich nur die Faktoren und als signifikant angesehene

Wechselwirkungen eingehen. Anstelle der höheren Wechselwirkungen werden in Teilfaktorplänen neue Faktoren eingesetzt, deren Niveauwerte mit denen der entsprechenden Wechselwirkungen übereinstimmen.
Im folgenden werden nur Funktionen zur Auswertung von vollständigen Faktorplänen behandelt. Allerdings ist es möglich, auch Teilfaktorpläne damit auszuwerten, indem man die Wechselwirkungen als Faktoren betrachtet. Dadurch erhält man formal einen vollständigen Faktorplan mit einem Grad, der niedriger ist als der Grad des vollständigen Faktorplans, der eigentlich zum verkürzten Versuchsplan gehört.

Von besonderer Bedeutung sind unter den faktoriellen Versuchsplänen die Zwei-Niveau-Pläne, da sie nicht nur weit verbreitet sind, sondern auch die obigen Forderungen an einen Versuchsplan erfüllen [8,48,51]. Über solche vollständigen Faktorpläne mit zwei Niveaus können die Koeffizienten von einer Modellgleichung ersten Grades

$$\eta = \beta_0 \cdot x_0 + \sum_{j=1}^{q} \beta_j \cdot x_j \tag{10-6}$$

oder einer degenerierten Modellgleichung zweiten Grades

$$\eta = \beta_0 \cdot x_0 + \sum_{j=1}^{q} \beta_j \cdot x_j + \sum_{i=1}^{q} \sum_{j=i+1}^{q} \beta_{ij} \cdot x_i \cdot x_j + \sum_{i=1}^{q} \sum_{j=i+1}^{q} \sum_{l=j+1}^{q} \beta_{ijl} \cdot x_i \cdot x_j \cdot x_l + \ldots \tag{10-7}$$

bestimmt werden. Mit der Funktion *edfulfac2n* werden diese Faktorpläne ausgewertet und die betrachteten Einflußgrößen auf ihre Signifikanz untersucht. Dazu werden unter anderem die Planmatrix, die Antworten, der Grad der Wechselwirkungen und eine Konfidenzzahl vorgegeben. Ferner besteht die Option, programmintern oder benutzerdefiniert die Auswahl an signifikanten Einflußgrößen zu treffen. (Vgl. Tab. 10-3)
Zur Feststellung signifikanter Einflußgrößen wird in *edfulfac2n* die Planmatrix gemäß Gl. (10-2) transformiert und die Berechnungsmatrix mit *edcalcm* aufgestellt (vgl. Kap. 10.1.1). Die Koeffizienten zur Modellgleichung Gl. (10-3) werden über den *backslash*-Operator vom MATLAB bestimmt (vgl. Kap. 6.1).
Um den Signifikanztest durchführen zu können, muß die Fehlervarianz bekannt sein. Da im allgemeinen keine Wiederholungsmessungen bei einem Versuchsplan vorliegen, wird die Fehlervarianz über eine Quadratsummenzerlegung programmintern abgeschätzt [30]. Bei der Quadratsummenzerlegung wird die gesamte Variation des Systems in einen erklärten und einen unerklärten Anteil

zerlegt. Für die Abschätzung der Fehlervarianz wird der Anteil $S\tilde{S}R$ der einzelnen Terme von Gl. (10-6) bzw. Gl. (10-7) an der erklärten Variation des transformierten Systems aufgestellt. Für den Quadratsummenanteil des *i*-ten Terms gilt

$$S\tilde{S}R_i = n \cdot \tilde{b}_i^{\,2} \tag{10-8}$$

Diese Teilquadratsummen entsprechen im vorliegenden Fall den unabhängigen Quadratsummenmittel $s_{R,i}^{\,2}$, aus denen man das mittlere Streuungsquadrat $s_R^{\,2}$ nach

$$s_R^{\,2} = \frac{1}{k} \cdot \sum_{i=1}^{k} s_{R,i}^{\,2} = \frac{1}{k} \cdot \sum_{i=1}^{k} S\tilde{S}R_i \tag{10-9}$$

erhält. Das mittlere Streuungsquadrat kann als Schätzwert der Fehlervarianz angesehen werden, wenn in der Summation in Gl. (10-9) nur *k* unabhängige Quadratsummenmittel von nicht-signifikanten Einflußgrößen auftreten. Dabei nimmt man an, daß die Koeffizienten von als vernachlässigbar zu betrachtenden Einflußgrößen nur durch den Versuchsfehler zustande gekommen und nicht auf einen wirklichen Effekt der Einflußgrößen zurückzuführen sind. Welche Quadratsummenmittel nicht signifikant sind, ist vor der Auswertung im allgemeinen nicht bekannt, weshalb in *edfulfac2n* die Entscheidung darüber optional vom Anwender oder programmintern über ein Iterationsalgorithmus getroffen wird.
Bei programminterner, iterativer Auswahl der nicht-signifikanten Quadratsummenmittel werden zunächst alle Teilquadratsummen in der Schätzung der Fehlervarianz berücksichtigt und ein kritischer Wert *M* nach BARTLETT [8,30] berechnet gemäß

$$M = k \cdot \ln\left(s_R^{\,2}\right) - \sum_{i=1}^{k} \ln\left(s_{R,i}^{\,2}\right) \tag{10-10}$$

Dieser *M*-Wert wird mit einem kritischen Wert nach NAIR verglichen [30,53], der ebenfalls von den beteiligten *k* Freiheitsgraden abhängt. Liegt *M* über diesem kritischen Wert, reduziert sich die Menge der betrachteten Quadratsummenmittel um das größte Element. Diese Prozedur wird fortgesetzt, bis der *M*-Wert unter dem entsprechenden kritischen Wert liegt. Die kritischen Werte nach NAIR werden über die Funktion *edfulfacnair* der Funktion *edfulfac2n* zugänglich gemacht.
Der eigentliche Signifikanztest besteht nun darin, daß für alle Faktoren und Wechselwirkungen ein *F*-Test durchgeführt wird [30]. Hierbei gelten alle

Faktoren und Wechselwirkungen als signifikant, für die bei einer gegebenen Konfidenzzahl γ gilt

$$s_{R,i}^{\ 2} > F_{\gamma}(1,k) \cdot s_R^{\ 2} \qquad (10\text{-}11)$$

Über die Funktion *edfulfac2n* erhält man die transformierten Koeffizienten der Modellgleichung, die Quadratsummenmittel, die rechten Seiten des *F*-Tests nach Gl. (10-11) und die Schätzung der Fehlervarianz sowie die Berechnungsmatrix (vgl. Tab. 10-3).

Tab. 10-3. Eigenschaften der Funktionen *edfulfac2n* und *edfulfacn*

edfulfac2n:	Auswertung eines vollständigen Faktorplans mit zwei Niveaus
Eingabe:	Planmatrix, Antwortvektor, Grad der gemischten Glieder*, Option auf programminterne oder benutzerdefinierte Signifikanzprüfung*, Konfidenzzahl*
Ausgabe:	transformierte Koeffizienten, Quadratsummenmittel, Vergleichszahlen aus *F*-Test, Schätzwert der Fehlervarianz, Berechnungsmatrix
edfulfacn:	Auswertung eines vollständigen Faktorplans mit *n* Niveaus
Eingabe:	Planmatrix, Antwortvektor, Grad der reinen Glieder, Grad der gemischten Glieder*, Option auf programminterne oder benutzerdefinierte Signifikanzprüfung*, Konfidenzzahl*
Ausgabe:	transformierte Koeffizienten, Quadratsummenmittel, Vergleichszahlen aus *F*-Test, Schätzwert der Fehlervarianz, Berechnungsmatrix, Berechnungsgleichungen für die Spalten der Berechnungsmatrix

* = Eingabe freigestellt

Mit der Funktion *edfulfacn* können Faktorpläne mit einer beliebigen Anzahl von Versuchsniveaus ausgewertet werden. Die Eigenschaften und Auswertungsgrundlagen stimmen weitgehend mit denen der Funktion *edfulfac2n* überein (vgl. Tab. 10-3). Unterschiede bestehen lediglich in der Aufstellung der Berechnungsmatrix und der Bestimmung der Quadratsummenmittel.

Bei Anwendung von *edfulfacn* wird die Berechnungsmatrix zunächst mit *edcalcm* aufgestellt (vgl. Kap. 10.1.1). Ist die erhaltene Matrix nicht orthogonal, wird die Matrix über *edcalcmo* berechnet. Erhält man wiederum eine nicht orthogonale Matrix, wird eine Warnung ausgegeben. Aufgrund dieser Problematik ist die Einsatzfähigkeit der Funktion *edfulfacn* eingeschränkt.

Im Gegensatz zu Gl. (10-8) geht in die Berechnung der Quadratsummenmittel nicht die Anzahl der Versuchspunkte, sondern das entsprechende Diagonalelement der aus der Berechnungsmatrix gebildeten Momentenmatrix ein [8].

10.2 Verfahren zur Parameterpräzisierung

Zur Parameterpräzisierung werden häufig Verfahren der Versuchsplanung eingesetzt. Dadurch werden Versuchspläne oder Versuchspunkte so festgelegt, daß auf ihrer Grundlage eine relativ genaue Bestimmung von Modellparametern möglich wird. Dabei liegen den Versuchsplänen im Gegensatz zu Kap. 10.1 im allgemeinen Modelle im Sinne von Gl. (7-1) und Gl. (8-1) zugrunde.
Der gesamte Versuchsplan besteht aus n Versuchen, die sich aus n_1 bereits durchgeführten und n_2 zu planenden Versuchen zusammensetzen. Hierbei umfaßt die Planmatrix eines solchen Versuchsplans die Einstellungen der n_1 Versuche und die Berechnungsmatrix die gesamten n Versuchseinstellungen (vgl. Kap. 10.1). Beide Matrizen unterscheiden sich demnach nur in der Anzahl ihrer Zeilen und nicht in der Anzahl der Spalten.
Um hierbei eine Parameterschätzung von p Parametern zu ermöglichen, muß folgende Bedingung für die Versuchsanzahl erfüllt sein

$$n = n_1 + n_2 \geq p \tag{10-12}$$

Als Kriterium für die Aufstellung eines optimalen Versuchsplans wird häufig die Forderung nach einem möglichst kleinen ellipsoiden gemeinsamen Vertrauensbereich der Modellparameter gewählt [9].
Dieser Sachverhalt wird mathematisch im sogenannten Box-Lucas-Kriterium ausgedrückt. Für Einfachantwortsysteme lautet es nach [9,54]

$$\det\left(\mathbf{J}^T \cdot \mathbf{J}\right) \rightarrow Max \tag{10-13}$$

In Gl. (10-13) stellt **J** die Jacobi-Matrix dar, wobei die Komponenten den Ableitungen der Modellgleichung nach den Parametern entsprechen. Für Mehrfachantwortsysteme kann das Box-Lucas-Kriterium unter Berücksichtigung der Varianz-Kovarianz-Matrix der Antworten **V** für r Modellgleichungen nach [9,55] folgendermaßen formuliert werden

$$\det\left(\sum_{i=1}^{r}\sum_{j=1}^{r} V_{ij}^{-1} \cdot \mathbf{J}_i^{\,T} \cdot \mathbf{J}_j\right) \rightarrow Max \tag{10-14}$$

Hierbei ist $\mathbf{J}_j$ die Jacobi-Matrix bezogen auf die j-te Modellgleichung. Die Bezeichnung V_{ij}^{-1} steht für die Komponente in der i-ten Zeile und j-ten Spalte der inversen Varianz-Kovarianz-Matrix. Für die Varianz-Kovarianz-Matrix wird auch hier angenommen, daß sie für alle Versuchspunkte gleich ist (vgl. Kap. 8).

Auf der Grundlage dieser Zielfunktionen kann die Erstellung der Versuchspläne entweder simultan oder sequentiell erfolgen.
Die Berechnung der Versuchspläne stellt ein Optimierproblem im Variablenraum dar, für dessen Behandlung in den folgenden Programmen das MATLAB-Optimierverfahren *fmins* in Verbindung mit der Möglichkeit zur Beschränkung des Variablenraums verwendet wird. Die Gründe für den Einsatz von *fmins* liegen ähnlich wie in Kap. 9.3.

10.2.1 Simultane Versuchsplanung

Verzeichnis: *exdesign*
Funktionen: *edsimult.m*
edsimultm.m

In der vorliegenden Programmsammlung kann die simultane Versuchsplanung für Einfachantwortsysteme mit der Funktion *edsimult* und für Mehrfachantwortsysteme mit *edsimultm* vorgenommen werden. Durch diese Funktionen werden Versuchspläne erstellt, bei denen die Versuchsanzahl der Anzahl an Parametern p entspricht. Die Planung von mehr Versuchen ist nicht sinnvoll, da sich in diesem Fall im allgemeinen die ersten p Versuche im Versuchsplan wiederholen [9].
Die besondere Problematik in der simultanen Versuchsplanung besteht darin, daß für die Anwendung des Box-Lucas-Kriteriums vermutete Schätzwerte bzw. Startschätzwerte für die Parameter (nach BOX engl. bezeichnet als „guestimates“) bekannt sein müssen (vgl. Kap. 12). Erst auf der Basis dieser Schätzwerte kann der optimale Versuchsplan simultan erstellt werden. Die Versuchsplanung mit nicht unbedingt gesicherten Modellparametern ist zulässig, da die Lage der optimalen Versuchspunkte im Variablenraum auch bei besser geeigneten Parameterschätzwerten nicht wesentlich verändert wird [9].
Für die Ermittlung von Startwerten und a-priori-Informationen können Vorversuche hilfreich sein.

Bei der Erstellung von Versuchsplänen mit den Funktionen *edsimult* und *edsimultm* müssen die Modellgleichung und die Parameterschätzwerte vorgegeben werden. Außerdem kann eine Beschränkung des Variablenraums vorgenommen werden. Für die eigentliche Optimierung ist die Vorgabe von Startwerten für den Versuchsplan nicht erforderlich. Diese werden gegebenenfalls intern eins gesetzt, wenn keine Begrenzung des Variablenbereichs vorliegt, oder berechnen sich jeweils als Mittelwert aus den vorgegebenen Intervallgrenzen der entsprechenden Variablen.

Ferner können bei *edsimult* und *edsimultm* optional a-priori-Informationen über die Modellparameter in Form der Varianz-Kovarianz-Matrix der Schätzung $\mathbf{V}_b$ und der Fehlervarianz σ^2 berücksichtigt werden, weshalb nicht das Kriterium nach Gl. (10-13) bzw. Gl. (10-14) verwendet wird, sondern eine modifizierte Form nach [9,56]. Für *edsimult* lautet das Schätzkriterium

$$\det\left(\mathbf{J}^T \cdot \mathbf{J} + \sigma^2 \cdot \mathbf{V}_b^{-1}\right) \rightarrow Max \tag{10-15}$$

und für *edsimultm*

$$\det\left(\sum_{i=1}^{r}\sum_{j=1}^{r} V^{ij} \cdot \mathbf{J}_i^T \cdot \mathbf{J}_j + \mathbf{V}_b^{-1}\right) \rightarrow Max \tag{10-16}$$

Liegen keine a-priori-Informationen bezüglich der Modellparameter vor, gehen die Gl. (10-15)...(10-16) in die Gl. (10-13)...(10-14) über.
Die hier aufgeführten Zielfunktionen sind für die Optimierung in den Routinen *edsimopt* bzw. *edsimmopt* abgelegt, von wo aus sie durch das Optimierverfahren aufgerufen werden. Die benötigten Jacobi-Matrizen werden über *jacsym* und *jacsymval* berechnet (vgl. Kap. 4.2). Bei Verwendung von *edsimultm* muß wegen Gl. (10-16) zusätzlich die Varianz-Kovarianz-Matrix bekannt sein.
Die Eigenschaften von *edsimult* und *edsimultm* befinden sich in Tab. 10-4.

Tab. 10-4. Eigenschaften der Funktionen zur simultanen Versuchsplanung

edsimult:	Simultane Versuchsplanung bei gleicher Anzahl von Versuchsniveaus und Parametern bei Einfachantwortsystemen
Eingabe:	Modellgleichung, Parameterschätzwerte, Startmatrix für den Versuchsplan*, Größe der Startmatrix*, zulässiger Variablenbereich*, Fehlervarianz* , Varianz-Kovarianz-Matrix der Schätzung*, Optimieroptionen*
Ausgabe:	Matrix des Versuchsplans, Jacobi-Matrix zum Versuchsplan, analytischer Ausdruck der Jacobi-Matrix zum Versuchsplan
edsimultm:	Simultane Versuchsplanung bei gleicher Anzahl von Versuchsniveaus und Parametern bei Mehrfachantwortsystemen
Eingabe:	Modellgleichungen, Parameterschätzwerte, Startmatrix für den Versuchsplan*, Größe der Startmatrix*, zulässiger Variablenbereich*, Varianz-Kovarianz-Matrix der Antworten , Varianz-Kovarianz-Matrix der Schätzung*, Optimieroptionen*
Ausgabe:	siehe *edsimult*

* = Eingabe freigestellt

10.2.2 Simultane Versuchsplanung bei unterschiedlichen Genauigkeitsanforderungen an die Modellparameter

Verzeichnis: *exdesign*
Funktionen: *edsimultp.m*

Für Einfachantwortsysteme können mit der Funktion *edsimultp* im Gegensatz zu *edsimult* Versuchspläne erstellt werden, die unterschiedliche Genauigkeitsanforderungen an einzelne Parameter qualitativ berücksichtigen. Durch die Verwendung von *edsimultp* werden Versuchspunkte so festgelegt, daß eine besonders genaue Bestimmung von ausgewählten Parametern ermöglicht wird.

Die Eingaben bei der Anwendung von *edsimultp* entsprechen weitgehend denen von *edsimult* (vgl. Kap. 10.2.1 u. Tab. 10-5).
Für die Berücksichtigung der unterschiedlichen Genauigkeitsanforderungen an die Parameter muß die Modellgleichung und der Parameterschätzvektor so aufgebaut sein, daß die ersten p_1 der p Komponenten genau zu ermitteln sind.
Für die Schätzung der Versuchspunkte wird nach [9] folgendes modifiziertes Box-Lucas-Kriterium betrachtet

$$\det(\mathbf{A} - \mathbf{B}^T \cdot \mathbf{C}^{-1} \cdot \mathbf{B}) \rightarrow Max \tag{10-17}$$

Dabei stellen die Matrizen **A**, **B** und **C** Untermatrizen der Matrix $\mathbf{J}^T\mathbf{J}$ aus Gl. (10-13) dar. Für diese Matrizen gilt

$$\mathbf{A} = \begin{bmatrix} (J^T J)_{1,1} & \cdots & (J^T J)_{1,p_1} \\ \vdots & \ddots & \vdots \\ (J^T J)_{p_1,1} & \cdots & (J^T J)_{p_1,p_1} \end{bmatrix} \quad \mathbf{B} = \begin{bmatrix} (J^T J)_{p_1+1,1} & \cdots & (J^T J)_{p_1+1,p_1} \\ \vdots & \ddots & \vdots \\ (J^T J)_{p,1} & \cdots & (J^T J)_{p,p_1} \end{bmatrix}$$
$$\mathbf{C} = \begin{bmatrix} (J^T J)_{p_1+1,p_1+1} & \cdots & (J^T J)_{p_1+1,p} \\ \vdots & \ddots & \vdots \\ (J^T J)_{p,p_1+1} & \cdots & (J^T J)_{p,p} \end{bmatrix} \tag{10-18}$$

Der gesamte Programmaufbau ist ähnlich zu *edsimult* (vgl. Kap. 10.2.1). Das Schätzkriterium nach Gl. (10-17) befindet sich in der Unterfunktion *edsimpopt*. Als Optimierverfahren wird ebenfalls *fmins* eingesetzt. Für die Berechnung der Jacobi-Matrizen werden *jacsym* und *jacsymval* verwendet (vgl. Kap. 4.2).
Die wesentlichen Eigenschaften von *edsimultp* sind in Tab. 10-5 aufgeführt.

Tab. 10-5. Eigenschaften der Funktion *edsimultp*

edsimultp:	Simultane Versuchsplanung bei gleicher Anzahl von Versuchsniveaus und Parametern bei Einfachantwortsystemen und unterschiedlichen Genauigkeitsanforderungen an die Parameter
Eingabe:	Modellgleichungen, Parameterschätzwerte, Startmatrix für den Versuchsplan*, Größe der Startmatrix*, zulässigen Variablenbereich*, Anzahl der genau zu bestimmenden Parameter*, Optimieroptionen*
Ausgabe:	siehe *edsimult* (Tab. 10-4)

* = Eingabe freigestellt

10.2.3 Sequentielle Versuchsplanung

Verzeichnis: *exdesign*
Funktionen: *edsequent.m*
edsequentm.m

Für eine sequentielle Versuchsplanung können die Funktionen e*dsequent* und *edsequentm* eingesetzt werden. Dabei wird *edsequent* bei Einfachantwortsystemen und *edsequentm* bei Mehrfachantwortsystemen verwendet (vgl. Tab. 10-6). Voraussetzung für die Anwendung dieser Funktionen ist allerdings, daß bereits n ($n > 1$) Vorversuche vorliegen. Mit diesen Funktionen wird zunächst der Bedarf an zusätzlichen Versuchen auf der Grundlage der experimentellen Gegebenheiten und der Vorversuche geprüft und anschließend gegebenenfalls ein weiterer (n+1)-ter Versuchspunkt geplant. Durch die vorgeschaltete Bedarfsprüfung wird untersucht, ob der Vertrauensbereich der Modellparameter durch weiteres Experimentieren eingeengt werden kann.

Tab. 10-6. Eigenschaften der Funktionen zur sequentiellen Versuchsplanung

edsequent:	Berechnung eines optimalen Versuchspunkts im Rahmen einer sequentiellen Versuchsplanung bei Einfachantwortsystemen
Eingabe:	Matrix der Einstellvariablen der Vorversuche, Modellgleichung, Parameterschätzwerte, relative Änderung des Versuchsplanungskriteriums, Schätzwert für den zu planenden Versuchspunkt*, zulässiger Variablenbereich*, Optimieroptionen*
Ausgabe:	neuer Versuchspunkt, relative Änderung des Versuchsplanungskriteriums, analytischer Ausdruck der Jacobi-Matrix, Jacobi-Matrix bezogen auf die n Vorversuche und den geplanten Versuch, Wert des Versuchsplanungskriteriums für (n+1), n und (n-1) Versuche

Forts. Tab.10-6.

edsequentm:	Berechnung eines optimalen Versuchspunkts im Rahmen einer sequentiellen Versuchsplanung bei Mehrfachantwortsystemen
Eingabe:	Matrix der Einstellvariablen der Vorversuche, Modellgleichung, Parameterschätzwerte, relative Änderung des Versuchsplanungskriteriums, Schätzwert für den zu planenden Versuchspunkt*, zulässiger Variablenbereich*, Varianz-Kovarianz-Matrix der Antworten, Optimieroptionen*
Ausgabe:	siehe *edsequent*

* = Eingabe freigestellt

Für den Einsatz dieser Funktionen müssen unter anderem die Matrix der Vorversuche, die Modellgleichungen und Parameterschätzwerte vorgegeben werden. Ferner kann man eine Beschränkung des zulässigen Variablenbereichs vornehmen oder Optimieroptionen abweichend von den MATLAB-Einstellungen setzen. Eine Vorgabe der Startschätzwerte für den zu berechnenden Versuchspunkt ist nicht zwingend notwendig, da diese intern mit eins belegt oder jeweils als Mittelwert aus den zulässigen Variablenintervallen bestimmt werden (vgl. Kap. 10.2.1). Bei Verwendung von *edsequentm* muß zusätzlich die Varianz-Kovarianz-Matrix der Fehler bekannt sein (vgl. Tab. 10-6).
Für die Bedarfsüberprüfung bezüglich der Planung neuer Versuche wird beiden Funktionen extern die experimentell bedingte, relative Änderung ε des Versuchsplanungskriteriums D vorgegeben, dabei muß für den Bedarf an weiteren Versuchen gelten

$$\frac{D^{(n)} - D^{(n-1)}}{D^{(n)}} > \varepsilon \qquad (10\text{-}19)$$

Der Index am Versuchsplanungskriterium D bezieht sich dabei auf die Anzahl der in der Berechnung des Kriteriums berücksichtigten Versuche. Als Versuchsplanungskriterium werden für *edsequent* Gl. (10-13) und für *edsequentm* Gl. (10-14) verwendet.
Bei einem festgestellten Bedarf an neuen Versuchen wird ein neuer Versuchspunkt auf der Grundlage der angegebenen Kriterien bestimmt. Für die Optimierung mit *fmins* sind die entsprechenden Versuchsplanungskriterien für *edsequent* in dem Unterprogramm *edseqopt* bzw. für *edsequentm* in *edseqmopt* abgelegt. Die Jacobi-Matrizen werden über *jacsym* und *jacsymval* berechnet.
Bei den Ausgabewerten der Funktionen *edsequent* und *edsequentm* handelt es sich um den neuen Versuchspunkt, den Wert des Bedarfskriteriums, den analytischen Ausdruck der Jacobi-Matrix, die Jacobi-Matrix für alle (n+1)

Versuche sowie die Werte des jeweiligen Versuchsplanungskriteriums für die (n+1), n und (n-1) Versuche (vgl. Tab. 10-6).

Der vollständige Ablauf der sequentiellen Versuchsplanung läßt sich folgendermaßen beschreiben:
Zunächst werden n Vorversuche durchgeführt. Auf der Grundlage dieser Versuche können dann die Modellparameter über die Funktionen aus Kap. 5 bis 8 geschätzt werden. Unter Berücksichtigung der Parameterschätzwerte und der n Versuche kann über *edsequent* bzw. *edsequentm* ein weiterer Versuchspunkt geplant werden. Nach Durchführung des neuen Versuchs werden die Parameter bezüglich der (n+1) Versuche erneut geschätzt. Über die Funktion *edsequent* bzw. *edsequentm* kann daraufhin wiederum der Bedarf an einem neuen Versuch geklärt und eventuell ein weiterer Versuch geplant werden. Diese zyklische Vorgehensweise wird wiederholt, bis vom Anwender die weitere Durchführung von Versuchen als nicht mehr zweckmäßig erachtet wird.
In diesem Ablauf einer sequentiellen Versuchsplanung übernehmen die Routinen *edsequent* bzw. *edsequentm* zwei Aufgaben, zum einen beinhalten sie mit der Bedarfsprüfung das Abbruchkriterium, zum anderen berechnen sie einen neuen, optimalen Versuchspunkt.

10.2.4 Anwendungsbeispiel zur Parameterpräzisierung

Anhand des folgenden Beispiels sollen einige der vorgestellten Verfahren zur Parameterpräzisierung bezüglich ihrer Einsatzfähigkeit untersucht werden. Beim dem Beispiel nach [9,57] handelt es sich um ein Simulationsexperiment zur Planung von geeigneten Versuchspunkten zur Parameterschätzung in einem Geschwindigkeitsgesetz, das die katalytische Dehydratisierung eines langkettigen primären Alkohols beschreibt. Das Geschwindigkeitsgesetz lautet in ybx-Schreibweise

$$y = \frac{b(1) \cdot b(3) \cdot x(1)}{1 + b(1) \cdot x(1) + b(2) \cdot x(2)} \qquad (10\text{-}20)$$

In Gl. (10-20) stellen $x(1)$ den Partialdruck und $b(1)$ die Adsorptionskonstante des Alkohols, $x(2)$ den Partialdruck und $b(2)$ die Adsorptionskonstante des Olefins und $b(3)$ die Bruttoreaktionsgeschwindigkeitskonstante dar.
Dieses Beispiel ist im folgenden sowohl mit den Möglichkeiten der simultanen als auch der sequentiellen Versuchsplanung bearbeitet worden.

Simultane Versuchsplanung

Für die Durchführung der simultanen Versuchsplanung sind die Parameterschätzwerte gemäß [9] mit

$$b(1) = 2.9 \qquad b(2) = 12.2 \qquad b(3) = 0.69 \tag{10-21}$$

angenommen und die Einstellbereiche der unabhängigen Variablen mit

$$0 \le x(1) \le 3 \qquad 0 \le x(2) \le 3 \tag{10-22}$$

festgelegt worden. Die Bestimmung des Versuchsplans erfolgte mit der Funktion *edsimult* unter Verwendung der Standardeinstellungen von *fmins* bei der Optimierung. Dabei sind nur zwei Versuchspunkte geplant worden, da analog zu [9] $b(3)$ als gesichert angesehen worden ist. Bei den Rechnungen sind a-priori-Informationen unberücksichtigt geblieben.

Als Startwert für die Planmatrix des Versuchsplans ist

$$\mathbf{X}_0 = \begin{pmatrix} 0.5 & 0 \\ 2 & 1 \end{pmatrix} \tag{10-23}$$

vorgegeben worden. Die durch *edsimult* erhaltenen Versuchspunkte sind in Tab. 10-7 den Literaturergebnissen gegenübergestellt. In Tab. 10-8 werden die berechneten Jacobi-Matrizen miteinander verglichen.

Tab. 10-7. Ergebnisse der simultanen Versuchsplanung am Beispiel der Alkoholdehydratisierung mit *edsimult* und nach [9]

Versuch	mit *edsimult*		nach [9]	
	$x(1)$	$x(2)$	$x(1)$	$x(2)$
1	0.3585	1.28*e*-7	0.3448	0
2	2.9992	0.8009	3.0	0.7951

Tab. 10-8. Jacobi-Matrizen **J** zu den Versuchsplänen aus Tab. 10-7

	mit *edsimult*	nach [9]
J	$\begin{pmatrix} 0.0595 & -2.21e{-8} \\ 0.0588 & -0.0127 \end{pmatrix}$	$\begin{pmatrix} 0.0595 & 0 \\ 0.0588 & -0.0127 \end{pmatrix}$

Die gute Übereinstimmung der Versuchspunkte und Jacobi-Matrizen mit den Literaturergebnissen verdeutlicht die korrekte programmtechnische Umsetzung

des Verfahrens zur simultanen Versuchsplanung in *edsimult*. Bei den Unterschieden in den Ergebnissen handelt es sich vermutlich um Rundungsfehler, die durch die jeweils eingesetzten numerischen Verfahren hervorgerufen werden.

Sequentielle Versuchsplanung

Bei der Anwendung der sequentiellen Versuchsplanung auf das Beispiel der Alkoholdehydratisierung soll zunächst die Vorgehensweise in [9] beschrieben werden:

Anfangs ist ein 2^2-Faktorplan vorgegeben. Die Antworten auf diesen Versuchsplan sind mit Hilfe der Modellgleichung Gl. (10-20) und den Parametern aus Gl. (10-21) berechnet und anschließend mit einem Zufallszahlengenerator mit einer Standardabweichung von 0.01 „verrauscht“ worden. Auf diese Vorversuche sind die Verfahren zur sequentiellen Versuchsplanung angewendet worden, d. h. es ist ein neuer Versuch geplant, nach dem obigen Prinzip die Antwort simuliert und schließlich der vollständige Parametersatz auf der Grundlage aller bisherigen Versuche neu geschätzt worden. Auf diese Weise sind insgesamt neun weitere Versuche simuliert und ausgewertet worden. Die Ergebnisse sind in Tab. 10-9 dargestellt.

Tab. 10-9. Literaturergebnisse der sequentiellen Versuchsplanung am Beispiel der Alkoholdehydratisierung nach [9]

n	*x*(1)	*x*(2)	*y*	*b*(1)	*b*(2)	*b*(3)
1	1.0	1.0	0.126			
2	2.0	1.0	0.219			
3	1.0	2.0	0.076			
4	2.0	2.0	0.126	10.39	48.83	0.79
5	0.1	0.0	0.186	3.11	15.19	0.66
6	3.0	0.0	0.606	3.96	15.32	0.66
7	0.2	0.0	0.268	3.61	14.00	0.67
8	3.0	0.0	0.614	3.56	13.96	0.67
9	0.3	0.0	0.318	3.32	13.04	0.67
10	3.0	0.8	0.298	3.33	13.48	0.63
11	3.0	0.0	0.509	3.74	13.71	0.63
12	0.2	0.0	0.247	3.58	13.15	0.63
13	3.0	0.8	0.319	3.57	12.77	0.63

In der Bearbeitung des Beispiels mit den Mitteln der vorliegenden Programmsammlung ist eine ähnliche Strategie wie in [9] eingesetzt worden. Für die Vorversuche ist auf denselben Faktorplan und dieselben Antworten zurückgegriffen worden, wie sie in Tab. 10-9 aufgeführt sind. Die Vorversuche sind ausgewertet, ein neuer Versuch geplant und die Antwort entsprechend obiger Beschreibung simuliert worden. Dieses ist ebenfalls neunmal wiederholt worden.
Für die Parameterschätzung ist die Funktion *multires* verwendet worden. Dabei ist *fmins* mit Standardeinstellungen als Optimierverfahren gewählt worden. Schätzkriterium war die ungewichtete Fehlerquadratsumme. Als Startschätzwerte haben jeweils die nach einer Auswertung erhaltenen Parameterschätzwerte gedient. Für die erste Schätzung sind die Startwerte nach Gl. (10-21) verwendet worden.
Die Versuchsplanung ist mit *edsequent* durchgeführt und das Optimierverfahren ist dabei ebenfalls in Standardeinstellungen angewendet worden. Die Beschränkung des zulässigen Variablenbereichs erfolgte gemäß Gl. (10-22).
Die Ergebnisse sind in Tab. 10-10 aufgeführt.

Tab. 10-10. Ergebnisse der sequentiellen Versuchsplanung am Beispiel der Alkoholdehydratisierung mit *edsequent*

n	*x*(1)	*x*(2)	*y*	*b*(1)	*b*(2)	*b*(3)
1	1.0	1.0	0.126			
2	2.0	1.0	0.219			
3	1.0	2.0	0.076			
4	2.0	2.0	0.126	7.1688	33.645	0.7444
5	2.9645	2.9999	0.1301	4.2991	20.228	0.7560
6	0.2410	0.0000	0.2922	2.3841	11.769	0.8009
7	0.0881	0.0000	0.1457	2.8442	12.313	0.7209
8	2.8366	0.0001	0.6129	3.0946	12.505	0.6831
9	2.9518	0.0000	0.6195	3.0759	12.490	0.6856
10	3.0000	0.7322	0.3305	3.0724	12.387	0.6860
11	3.0000	0.7336	0.3272	3.0727	12.394	0.6860
12	2.9816	0.6366	0.3404	3.0779	12.517	0.6854
13	2.8662	0.0000	0.6176	3.0719	12.519	0.6862

Die hier mit *edsequent* und *multires* erhaltenen Ergebnisse zeigen, daß die umgesetzten Verfahren zur sequentiellen Versuchsplanung und Parameter-

schätzung durchaus die Erwartungen erfüllen, indem die Literaturergebnisse in ihrem Verlauf und ihrer Größenordnung bestätigt werden (vgl. Tab. 10-9 u. Tab. 10-10). Die Unterschiede zu den Literaturergebnissen haben ihre Ursache höchstwahrscheinlich in den jeweils verwendeten numerischen Verfahren und der Genauigkeit, mit der die Zwischenergebnisse im sequentiellen Ablauf weiterverarbeitet worden sind. Ferner ist in diesem Zusammenhang auch die Verwendung von zufällig „verrauschten“ Antworten zu berücksichtigen.

Anhand beider Versuchspläne läßt sich erkennen, daß es sich nach Durchführung von sechs Versuchen bei den neuen Versuchspunkten lediglich um Wiederholungen der ersten Versuche handelt. Diesbezüglich wiederholen sich in den Literaturergebnissen die Versuchspunkte (0.2;0.0), (3.0;0.0) und (3.0;0.8), während in Tab. (10-10) nach dem sechsten Versuch hauptsächlich die Versuchspunkte (2.8/2.9;0.0) und (2.99/3.00;0.6/0.7) auftreten. Dieses kann als Bestätigung der in Kap. 10.2.1 getroffenen Aussage betrachtet werden, nach der lediglich so viele Versuche geplant werden müssen, wie Modellparameter zu bestimmen sind, da sich darüber hinaus geplante Versuche bezüglich der Einstellvariablen wiederholen.

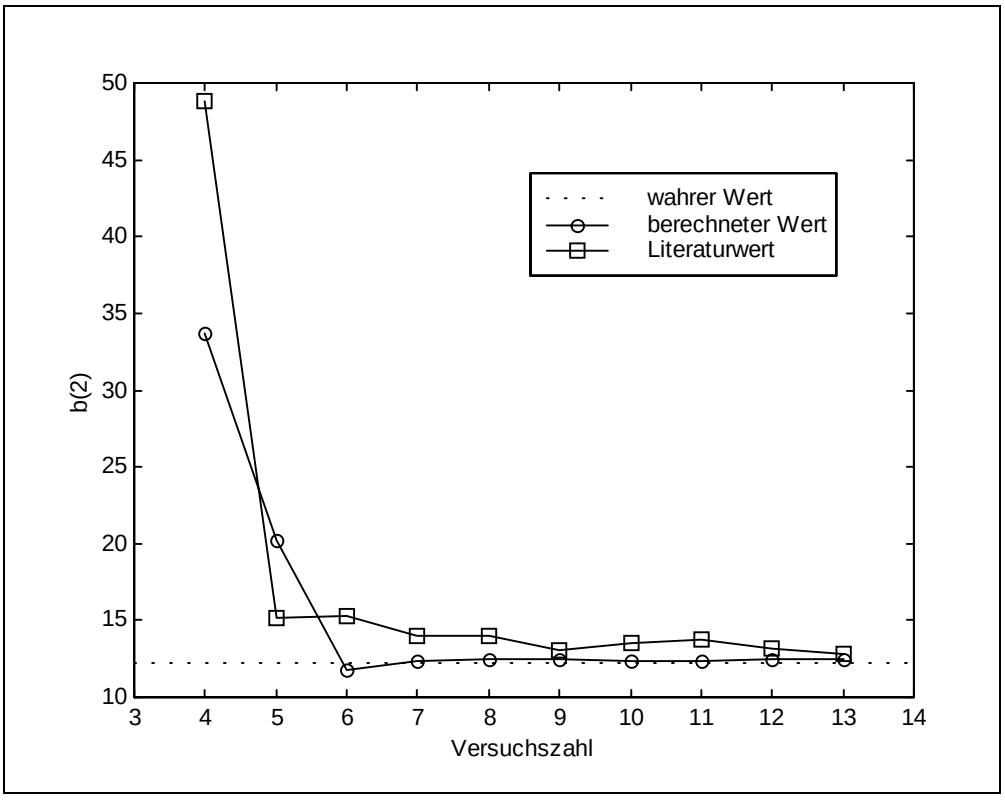

Abb. 10-1. Darstellung des Verlaufs der Parameterschätzwerte für $b(2)$ in Abhängigkeit der Versuchszahl und verschiedener Verfahren mit MATLAB

Außerdem läßt sich für das vorliegende Beispiel feststellen, daß die mit den Funktionen *edsequent* und *multires* an jedem Versuchspunkt erhaltenen Parameterschätzwerte näher an den tatsächlichen Werten gemäß Gl. (10-21) liegen als die entsprechenden Literaturergebnisse. Dieses kann als Hinweis auf die Leistungsfähigkeit von MATLAB und der vorliegenden Programme betrachtet werden. Um diese Aussagen zu verdeutlichen, ist in Abb. 10-1 der Verlauf der Parameterwerte von $b(2)$ für die unterschiedlichen Versuchspläne dargestellt.

Zusammenfassend kann festgestellt werden, daß die am vorliegenden Beispiel der Alkoholdehydratisierung überprüften Verfahren zur simultanen und sequentiellen Versuchsplanung grundsätzlich bei der Parameterpräzisierung einsetzbar sind, was unter anderem durch die gute Übereinstimmung mit den Literaturergebnissen bestätigt wird. Die Leistungsfähigkeit dieser Routinen hängt jedoch maßgeblich von den verwendeten MATLAB-Verfahren ab.

Um ein hohes Maß an Vertrauen in die geschätzten Modellparameter zu erreichen, empfiehlt es sich, die Vorversuche über ein Verfahren der simultanen Versuchsplanung zu ermitteln und anschließend die Parameterschätzwerte durch ein sequentielles Vorgehen zu präzisieren. Der Vorteil der Kombination beider Verfahren besteht darin, daß durch die simultane Versuchsplanung geeignete Vorversuche für ein sequentielles Vorgehen bestimmt werden können. Zudem erhöht das Wechselspiel zwischen Experiment und Auswertung ständig den Informationsgehalt für die Parameterschätzung und Versuchsplanung, wodurch ein gezieltes und schnelles Vorgehen für eine genaue Parameterermittlung möglich wird.

11 Parameterschätzung in Systemen gewöhnlicher Differentialgleichungen

In der Chemie betrachtet man üblicherweise Reaktionen in geschlossenen Systemen. Zur Beschreibung der zeitlichen Verläufe der Stoffmengen oder Konzentrationen der Reaktionsteilnehmer werden Differentialgleichungen herangezogen. Für eine Reaktionskomponente *i* gilt nach [24]

$$R_i = \frac{dn_i}{Vdt} = f_i(\mathbf{k},\mathbf{c}) = h_i(\boldsymbol{\beta},\boldsymbol{\eta}) \tag{11-1}$$

Dabei ist R_i die Stoffmengenänderungsgeschwindigkeit, die sich als Funktion der Konzentrationen der einzelnen Komponenten c_i und der Geschwindigkeitskonstanten der einzelnen Reaktionen k_j formulieren läßt.
Für eine vollständige Beschreibung sind neben den Stoffbilanzen gegebenenfalls noch weitere Bilanzgleichungen, wie z. B. die Enthalpiebilanzen, zu berücksichtigen.

Die in diesem Abschnitt betrachteten Differentialgleichungssysteme können mit der Integrationsvariablen *x* folgendermaßen formuliert werden

$$\boldsymbol{\eta}' = \mathbf{h}(\boldsymbol{\beta},\boldsymbol{\eta},x) \tag{11-2}$$

In diesem Zusammenhang kann die für die Berechnung der Vertrauensintervalle und Gradienten notwendige Jacobi-Matrix nicht mehr über die Routine *jacsym* aufgestellt werden, da die integralen Modellgleichungen im allgemeinen nicht zur Verfügung stehen (vgl. Kap. 4.2). Die Komponenten der Jacobi-Matrix werden deshalb nach [7] aus den Sensitivitätsgleichungen ermittelt

$$\frac{d}{dt}\left(\frac{\partial\boldsymbol{\eta}}{\partial\boldsymbol{\beta}}\right) = \frac{\partial\mathbf{h}}{\partial\boldsymbol{\beta}} + \frac{\partial\mathbf{h}}{\partial\boldsymbol{\eta}}\cdot\frac{\partial\boldsymbol{\eta}}{\partial\boldsymbol{\beta}} \tag{11-3}$$

Im folgenden werden zwei Verfahren zur Parameterschätzung in Differentialgleichungen vorgestellt, denen die numerische Integration und Optimierung zugrunde liegt, aber die sich hinsichtlich der Berechnung des Gradienten der Schätzfunktion und der Anzahl der zu schätzenden Parameter unterscheiden.
Für die Behandlung von Differentialgleichungen verfügt MATLAB über eine Auswahl an verschiedenen Integrationsverfahren, die sich im Rahmen dieser

Arbeit als sehr leistungsstark und allgemein einsetzbar erwiesen haben, weshalb auf die Implementierung spezieller Lösungsverfahren verzichtet worden ist.
Auf die Verwendung weiterer Methoden wie der Quasilinearisierung der Modellgleichungen bezüglich der Parameter unter Anwendung eines Newton-Rhaphson-Verfahrens nach [52] ist verzichtet worden, da die Ergebnisse im Vergleich zu den implementierten Verfahren unbefriedigend waren.

Um die in diesem Abschnitt diskutierten Programme anwenden zu können, müssen beim Aufstellen der Differentialgleichungen folgende Anforderungen erfüllt werden:

- Die Gleichungssysteme müssen vollständig sein. Dieses bedeutet unter anderem, daß für alle beteiligten Komponenten Stoffbilanzen aufgestellt werden müssen. Man kann das System jedoch sinnvoll auf die Schlüsselkomponenten reduzieren, wobei die Meßinformationen durch Nicht-Schlüsselkomponenten gegebenenfalls berücksichtigt werden sollten [7].
- Analog zu den vorherigen Abschnitten wird auch hier eine ybx-Schreibweise verwendet. Die zu integrierenden Größen werden mit *y* und die Parameter mit *b* bezeichnet. Bei den *y*-Werten kann es sich um Konzentrationen, Stoffmengen, Temperaturen oder Volumina handeln. Die Zeit wird durch *x* repräsentiert. Alle anderen Größen müssen in Form von Zahlenwerten in die Gleichungen eingesetzt werden.
- Sind nicht für alle notwendigen Größen die erforderlichen Meßdaten vorhanden, so müssen die fehlenden Daten gegebenenfalls in der Antwortmatrix mit dem MATLAB-Ausdruck *NaN* belegt werden (vgl. Kap. 8.4).

11.1 Parameterschätzung mit numerische r Integration

Verzeichnis: *dynmodels*
Funktionen: *dmnumint.m*

Die Parameterschätzung in gewöhnlichen Differentialgleichungen kann über die Funktion *dmnumint* durchgeführt werden. Dazu werden die Differentialgleichungen mit den von MATLAB zur Verfügung gestellten Verfahren integriert. Die so erhaltenen Funktionswerte werden anschließend in den Optimierungsablauf eingebunden, der analog zu *multires* verläuft (vgl. Kap. 8.2). Die Schätzkriterien sind dabei in der Routine *dmobjfcn* abgelegt und entsprechen den Funktionen in Tab. 8-1. Als zusätzliche Option können mit dieser Routine auch Modellwerte über den Umfang der Meßstellen hinaus ermittelt werden.

Ferner besteht bei Anwendung von *dmnumint* wahlweise die Möglichkeit, die Anfangswerte der Differentialgleichungen ebenfalls als Schätzparameter in den Optimierungsprozeß einzubinden. Um den Rechenaufwand dabei möglichst gering zu halten, werden nur Anfangswerte als Schätzparameter behandelt, die nicht Null sind. Dieses Vorgehen ist gerechtfertigt, da bei der Betrachtung chemischer Systeme Stoffmengen anfänglich nicht vorhandener Substanzen jeweils exakt Null sind.
Als Optimierverfahren sind *fmins* und gegebenenfalls aus der *Optimization Toolbox* von MATLAB *leastsq* und *constr* vorgesehen.
Die Eigenschaften und Anforderungen von *dmnumint* sind in Tab. 11-1 aufgeführt.

Tab. 11-1. Eigenschaften der Funktion *dmnumint*

dmnumint:	Parameterschätzung in Systemen gewöhnlicher Differentialgleichungen
Eingabe:	Vektor der Integrationsstellen, Antwortmatrix, Modellgleichungen, Startschätzwerte der Parameter, Anfangswerte der Integration, Vektor zusätzlicher Integrationsstellen*, Wahl des DGL-Lösers*, Integrationsoptionen*, Optimieroptionen*, Optimierverfahren*, Schätzfunktion*, Varianz-Kovarianz-Matrix*, zulässiger Parameterbereich*, Nebenbedingungen*, Option zur Schätzung der Anfangswerte*
Ausgabe:	Parameterschätzwerte, Wert des Schätzkriteriums, Residuen, Modellwerte

* = Eingabe freigestellt

Die Programmstruktur ist wegen der Verwendung von Integrationsverfahren, sogenannten DGL-Lösern, umfangreicher als bei der vergleichbaren Routine *multires* (vgl. Kap. 8.2).
Das Hauptprogramm *dmnumint* regelt die Eingabe, Ausgabe und Weitergabe von Informationen. Von dem Hauptprogramm wird über die Routine *optrout* das Optimierverfahren aufgerufen, welches wiederum auf das Programm *dmniopt* zurückgreift. In *dmniopt* wird das gewählte Schätzkriterium unter Verwendung von *dmobjfcn* berechnet und der DGL-Löser zur Ermittlung der Modellwerte aufgerufen. Der DGL-Löser integriert hierbei die an die Datei *dmniode* übermittelten Differentialgleichungen. Nach erfolgreicher Optimierung werden im Hauptprogramm nochmals alle entscheidenden Größen wie der Wert des Schätzkriteriums, die Residuen und die Modellwerte bestimmt. Dazu wird im Hauptprogramm eine Integration durchgeführt und *dmobjfcn* aufgerufen. Dieser zusätzliche, wiederholte Rechenschritt ist notwendig, um die Verwendung von global definierten Variablen zu vermeiden (vgl. Kap. 3.5.1).

Die Programmstruktur von *dmnumint* ist in Verbindung mit den Unterprogrammen in Abb. 11-1 dargestellt.

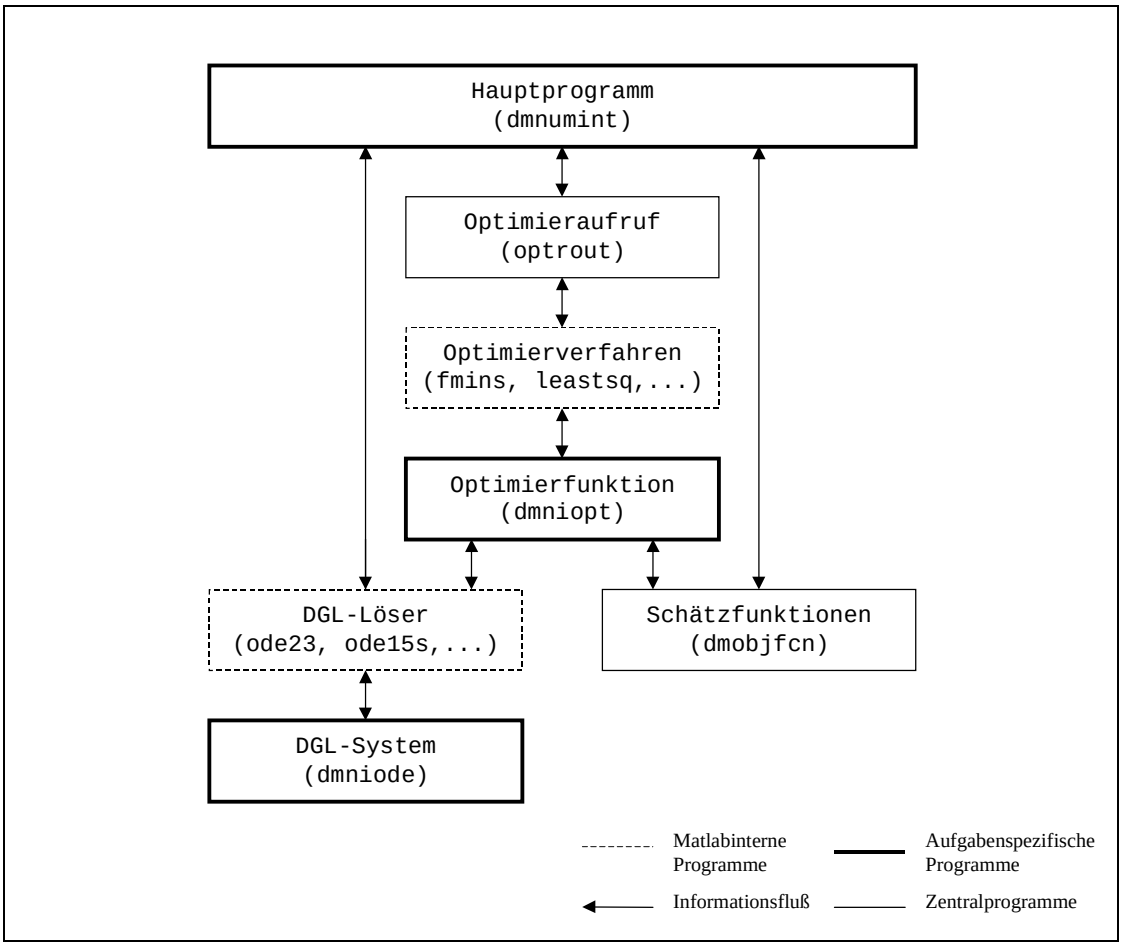

Abb. 11-1. Struktureller Aufbau der Programme zur Parameterschätzung in gewöhnlichen Differentialgleichungen

11.2 Parameterschätzung unter Berücksichtigung der Sensitivitätskoeffizienten

Verzeichnis: *dynmodels*
Funktionen: *dmseneq.m*

Das Programmpaket *dmseneq* ist ähnlich aufgebaut wie *dmnumint*, allerdings kann hier der Gradient der Bewertungsfunktion vorgegeben werden, indem aus Gl. (11-3) die Sensitivitätskoeffizienten und hieraus wiederum die Jacobi-Matrix ermittelt werden. Für den Gradienten **q** der Bewertungsfunktion bei *n* Versuchsniveaus gilt nach [7]

$$\mathbf{q} = -2 \cdot \sum_{i=1}^{n} \frac{\partial \Phi}{\partial \mathbf{e}_i} \cdot \frac{\partial \hat{\mathbf{y}}_i}{\partial \mathbf{b}} = -2 \cdot \sum_{i=1}^{n} \mathbf{J}_i^T \cdot \boldsymbol{\Gamma} \cdot \mathbf{e}_i \qquad (11\text{-}4)$$

Hierbei stehen **e** für den Vektor der Residuen, **Γ** für die Ableitungsmatrix der Bewertungsfunktion nach der Fehlermatrix **D** und Φ für das Bewertungskriterium. Die funktionalen Zusammenhänge für **Γ** können für die voreingestellten Schätzkriterien Tab. 8-1 entnommen werden.
Der Vorteil dieser Methode besteht darin, daß der Gradient mit der gleichen Genauigkeit wie die Bewertungsfunktion selbst berechnet werden kann.
In seinem Einsatzbereich ist *dmseneq* jedoch eingeschränkt, denn die Berücksichtigung der Gradientenfunktion ist bisher nur bei Verwendung von *leastsq* und *constr* als Optimierroutinen möglich, was die *Optimization Toolbox* voraussetzt [18] (vgl. Kap. 3.5.1).
Die zweite Einschränkung besteht darin, daß dieses Programm nicht mit fehlenden Meßdaten umgehen kann, da die Anwendung der *NaN*-Funktion in der Gradientenberechnung nicht möglich ist.
Die Ein- und Ausgaben der Funktion *dmseneq* entsprechen weitgehend den Argumenten von *dmnumint* (vgl. Tab. 11-2) Zusätzlich müssen jedoch Angaben zu den Anfangswerten der Sensitivitätsgleichungen gemacht werden. Im Unterschied zu *dmnumint* kann die Schätzung von Anfangswerten nicht vorgenommen werden. Bleibt die Option zur Berechnung des Gradienten der Schätzfunktion unberücksichtigt, entspricht *dmseneq* im internen Programmablauf *dmnumint.*

Tab. 11-2. Eigenschaften der Funktion *dmseneq*

dmseneq:	Parameterschätzung in Systemen gewöhnlicher Differentialgleichungen unter Berücksichtigung der Sensitivitätsgleichungen
Eingabe:	Vektor der Integrationsstellen, Antwortmatrix, Modellgleichungen, Startschätzwerte der Parameter, Anfangswerte der Modellgleichungen, Anfangswerte der Sensitivitätsgleichungen, Vektor zusätzlicher Integrationsstellen*, Wahl des DGL-Lösers*, Integrationsoptionen*, Optimieroptionen*, Optimierverfahren*, Schätzfunktion*, Varianz-Kovarianz-Matrix*, zulässiger Parameterbereich*, Nebenbedingungen* (bei *constr*)
Ausgabe:	Parameterschätzwerte, Wert des Schätzkriteriums, Residuen, Modellantworten, Sensitivitätskoeffizienten und ihre analytische Form

* = Eingabe freigestellt

Der Programmaufbau von *dmseneq* ist wegen der Berechnung des Gradienten zur Schätzfunktion umfangreicher als bei *dmnumint* (vgl. Abb. 11-1 u. 11-2).

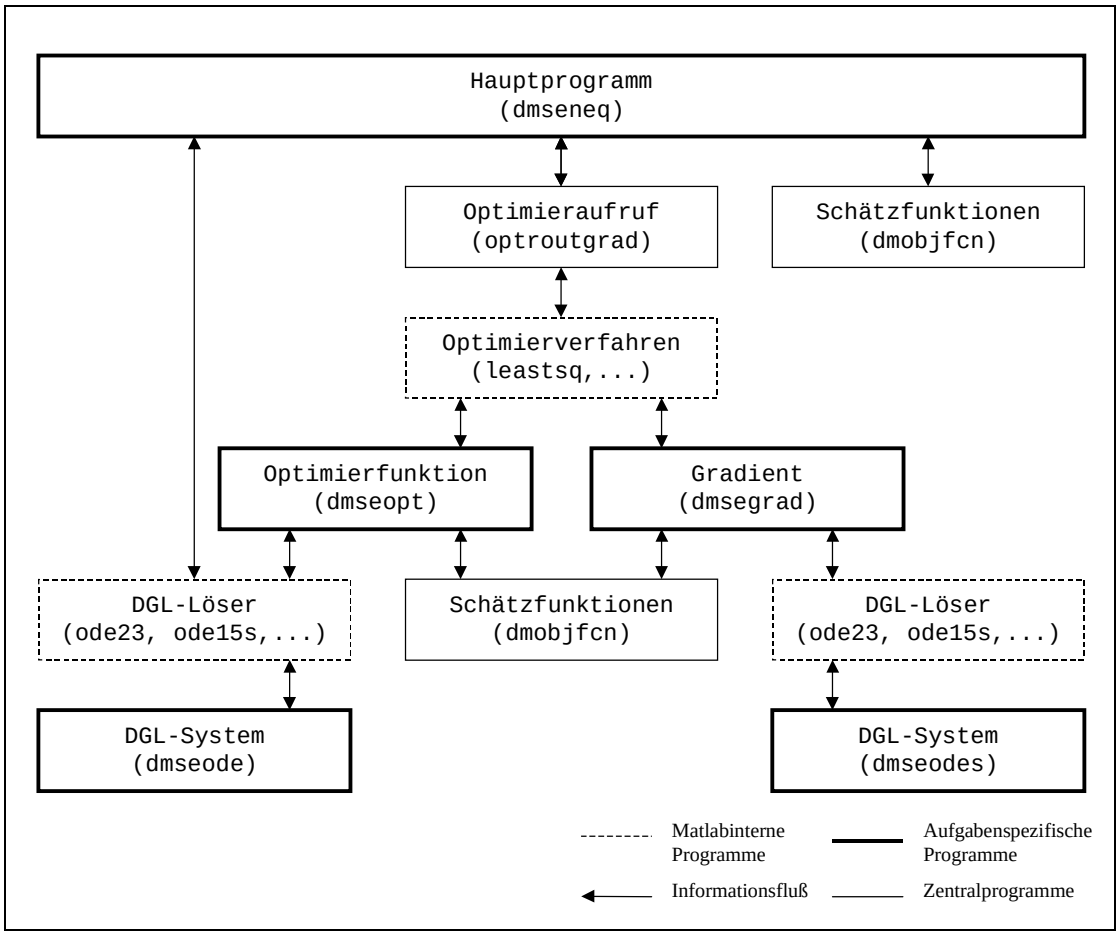

Abb. 11-2. Struktureller Aufbau der Programme zu Parameterschätzung in Differentialgleichungssystemen unter Berücksichtigung der Sensitivitätsgleichungen

In *dmseneq* erfolgt der Aufruf des Optimierverfahrens über *optroutgrad*, um so die Möglichkeit zu haben, den Gradienten der Bewertungsfunktion extern vorzugeben (vgl. Kap. 4.2). Die Funktion *optroutgrad* ruft nicht nur das Optimierverfahren auf, sondern übermittelt diesem, daß die Aufstellung des Gradienten in der Routine *dmsegrad* erfolgt. In der Funktion *dmsegrad* werden die Sensitivitätskoeffizienten gemäß Gl. (11-3) nicht bei jedem Rechenschritt über Differenzenbildung bestimmt, sondern statt dessen programmintern die analytischen Ausdrücke aufgestellt, als *string* formuliert und mit den Modellgleichungen zusammen verarbeitet. Neben dem DGL-Löser kommt hierbei die Funktion *dmseodes* zum Einsatz, in der sich das vollständige Differentialgleichungssystem aus Sensitivitäts- und Modellgleichungen befindet. Der übrige Ablauf der Parameterschätzung erfolgt analog zur Funktion *dmnumint* (vgl. Kap. 11.1).
Aus Gründen der Übersichtlichkeit sind in Abb. 11-2 die DGL-Löser und Schätzfunktionen jeweils zweimal aufgeführt, obwohl jedesmal die gleichen Verfahren eingesetzt werden.

11.3 Vertrauensbereiche der Parameter

Verzeichnis: *dynmodels*
Funktionen: *dmconfp.m*
dmcorrp.m

Die individuellen Vertrauensbereiche der Parameter werden über die Funktion *dmconfp* berechnet. Im einzelnen werden auch hier die Varianz-Kovarianz-Matrix der Schätzung, die Standardabweichungen, die Konfidenzintervalle, die Tangentialebenenvertrauensintervalle, der Wert des Schätzkriteriums, der Gradient, die Jacobi-Matrix und die Hesse-Matrix sowie die analytischen Darstellungen der Sensitivitätsgleichungen erhalten. Die Vorgaben sind analog zu *dmseneq* (vgl. Kap. 11.2), da für die Berechnung der Vertrauensintervalle die Jacobi-Matrix ebenfalls über die Sensitivitätsgleichungen aufgestellt wird. Dieses Vorgehen hat sich im Vergleich zu einer Bestimmung der Jacobi-Matrix über Differenzenbildung als sehr effektiv erwiesen (vgl. Kap. 4.2). Die wesentlichen Eigenschaften von *dmconfp* sind in Tab. 11-3 zusammengefaßt.

Tab. 11-3. Eigenschaften der Funktionen *dmconfp und dmcorrp*

dmconfp:	Bestimmung individueller linearisierter Vertrauensintervalle von Parametern in Systemen gewöhnlicher Differentialgleichungen
Eingabe:	Vektor der Integrationsstellen, Antwortmatrix, Modellgleichungen, Startschätzwerte der Parameter, Anfangswerte der Modellgleichungen, Anfangswerte der Sensitivitätsgleichungen, Vektor zusätzlicher Integrationsstellen*, Wahl des DGL-Lösers*, Integrationsoptionen*, Schätzfunktion*, Varianz-Kovarianz-Matrix der Antworten*
Ausgabe:	Parameterschätzwerte, Varianz-Kovarianz-Matrix der Schätzung, Standardabweichung, Konfidenzintervallhalblängen, Halblängen des Tangentialebenenvertrauensintervalls, Wert der Schätzfunktion, Gradient, Jacobi-Matrix, Hesse-Matrix (jeweils bezogen auf die Parameter)
dmcorrp:	Darstellung des konturgetreuen gemeinsamen Vertrauensbereichs von zwei ausgewählten Parametern in Systemen gewöhnl. Differentialgleichungen
Eingabe:	Vektor der Integrationsstellen, Antwortvektor, Schätzvektor der Parameter, Modellgleichungen, graphisch darzustellende Parameter, zulässiger Parameterbereich*, Konfidenzzahl*
Ausgabe:	Matrix der Niveaupunkte, Parameterwerte

* = Eingabe freigestellt

Der Programmablauf in der Funktion *dmconfp* ist ähnlich zu *srconfp* und *mrconfp* (vgl. Kap. 7.4 u. 8.2). Über Gl.(7-9)...(7-11) werden der Gradient, die

Hesse-Matrix und Varianz-Kovarianz-Matrix der Schätzung und über Gl. (6-7) und Gl.(7-7) die entsprechenden individuellen Vertrauensintervalle berechnet.

Für die graphische Darstellung und Ermittlung des konturgetreuen gemeinsamen Vertrauensbereichs zwischen jeweils zwei ausgewählten Parametern kann die Funktion *dmcorrp* eingesetzt werden. Bei Anwendung von *dmcorrp* wird wie bei den analogen Programmen über den Parameterraum ein Gitter gelegt und die Matrix für die Niveaulinie des Schätzkriteriums mit dem Höhenplotbefehl *contour* berechnet (vgl. Kap. 7.4 u. 8.3). Dabei wird der Vertrauensbereich jedoch relativ ungenau dargestellt, denn bei Verwendung von *dmcorrp* wird kein Bezug auf die ausgewählte Schätzfunktion genommen, sondern standardmäßig die ungewichtete Fehlerquadratsumme als Kriterium gewählt (vgl. Tab. 11-2). Außerdem werden nicht wie üblich die Funktionswerte, sondern ihre durch Differenzenbildung ermittelten Ableitungen verglichen (vgl. Kap. 6,7,8). Wegen der Betrachtung der Ableitungen bei der Berechnung der Niveaulinien ist das Verfahren zwar sehr ungenau, aber ein Vorgehen über die Funktionswerte würde an jedem Gitterpunkt eine Integration verlangen. Diese Integrationen wirken sich jedoch in erheblichem Maße auf den Rechenaufwand und die Speicherkapazität aus. Es empfiehlt sich daher, beim Vorliegen der analytischen Lösungen der Differentialgleichungen die bereits vorgestellten Programme für die Berechnung gemeinsamer Vertrauensintervalle bei Mehrfachantwortsystemen zu verwenden (vgl. Kap. 8.3).

11.4 Anwendungsbeispiel

Die Einsatzfähigkeit von einigen der hier vorgestellten Funktionen soll an einem in [7] gut untersuchten Beispiel dargestellt werden.
Dabei wird folgende katalysierte Gleichgewichtsreaktion betrachtet

$$\mathrm{A} \rightleftharpoons 2\,\mathrm{B} \tag{11-5}$$

Die Reaktion genüge dem Geschwindigkeitsgesetz

$$r = \frac{k_1 \cdot \left(c_A - K \cdot c_B^2\right)}{\left(1 + k_2 \cdot c_A\right)^2} = \frac{k_{01} \cdot \exp(-E_{a1}/RT) \cdot \left(c_A - \exp(-1000/T) \cdot c_B^2\right)}{\left(1 + k_{02} \cdot \exp(-E_{a2}/RT) \cdot c_A\right)^2} \tag{11-6}$$

Die Geschwindigkeitskonstanten k_1 und k_2 sowie die Gleichgewichtskonstante K sind in Gl. (11-6) durch die entsprechenden Exponentialterme ersetzt worden. Es sollen in dem Modell die Größen k_{01}, E_{a1}, k_{02} und E_{a2} geschätzt werden

Für die Konzentrationsänderungen gelte

$$\frac{dc_A}{dt} = -r \quad , \quad \frac{dc_B}{dt} = 2r \tag{11-7}$$

Diesem Beispiel sind drei Versuchsreihen bei den Temperaturen 200 K, 400 K und 600 K zugrunde gelegt worden.
Bei der Parameterschätzung sind die Versuchsreihen simultan ausgewertet worden, indem jede Meßreihe einer Komponente bei einer bestimmten Temperatur als eigenständige Antwort des System betrachtet worden ist.
Bei der Aufstellung der Modellgleichungen in der ybx-Schreibweise sind die Temperaturen als konstante Größen aufgefaßt worden. Für T = 200 K stellen sich die Stoffmengenänderungsgeschwindigkeiten in ybx-Schreibweise folgendermaßen dar

$$\begin{aligned}\frac{dy(1)}{dt} &= -\frac{b(1)\cdot\exp(-b(2)/200)\cdot\left(y(1)-\exp(-1000/200)\cdot y(2)^2\right)}{\left(1+b(3)\cdot\exp(-b(4)/200)\cdot y(1)\right)^2}\\ \frac{dy(2)}{dt} &= 2\frac{b(1)\cdot\exp(-b(2)/200)\cdot\left(y(1)-\exp(-1000/200)\cdot y(2)^2\right)}{\left(1+b(3)\cdot\exp(-b(4)/200)\cdot y(1)\right)^2}\end{aligned} \tag{11-8}$$

Entsprechend zu Gl. (11-8) sind die Stoffmengenänderungsgeschwindigkeiten zu den übrigen Temperaturen formuliert worden. Der Stoff A wird hierbei für die unterschiedlichen Temperaturen durch die Komponenten 1, 3, und 5 des Antwortvektors ausgedrückt, während B durch die Komponenten 2, 4 und 6 repräsentiert wird.

Bei der Durchführung der Parameterschätzung ist über alle Meßstellen der drei Versuchsreihen integriert worden. Da nicht an jeder Meßstelle eine Messung durchgeführt worden ist, sind die Fehlstellen in der Antwortmatrix durch die *NaN*-Funktion von MATLAB ersetzt worden. Die Meßwerte sind in Tab. 11-4 aufgeführt.

Tab. 11-4. Meßdatensatz für die Gleichgewichtsreaktion nach Gl. (11-5)

	T = 200 K		T = 400 K		T = 600 K	
t (in s)	$y(1)$	$y(2)$	$y(3)$	$y(4)$	$y(5)$	$y(6)$
0	1.00830	0.99662	0.98862	0	0	1.01731
0.5	–	–	–	–	0.01688	0.98263
1	–	–	–	–	0.03331	0.94991
1.5	–	–	–	–	0.04726	0.92359
2	–	–	0.93445	0.10857	0.05604	0.90485
2.5	–	–	–	–	0.06330	0.89059
3	–	–	–	–	0.07073	0.87613
3.5	–	–	–	–	0.07811	0.86138
4	–	–	0.88192	0.21395	0.08214	0.85298
4.5	–	–	–	–	0.08782	0.84250
6	–	–	0.82997	0.31716	–	–
8	–	–	0.78015	0.41627	–	–
10	0.98040	1.05134	0.73047	0.51561	–	–
12	–	–	0.68271	0.61219	–	–
14	–	–	0.63816	0.70085	–	–
16	–	–	0.59445	0.78925	–	–
18	–	–	0.55375	0.86976	–	–
20	0.95262	1.10796	–	–	–	–
30	0.92703	1.16017	–	–	–	–
40	0.90120	1.21002	–	–	–	–
50	0.87706	1.26056	–	–	–	–
60	0.84883	1.31534	–	–	–	–
70	0.82480	1.36299	–	–	–	–
80	0.80163	1.41035	–	–	–	–
90	0.77726	1.45822	–	–	–	–

Für die Ermittlung der Varianzen der einzelnen Antworten sind gemäß [7] ausgeglichene Meßwerte $\bar{y}_j(i)$ an den Versuchspunkten j erzeugt worden. Für die Berechnung dieser Meßwerte gilt bei T = 200 K in Abhängigkeit der Anfangswerte $y_0(i)$

$$\bar{y}_j(1) = \frac{1}{5} \cdot y_j(1) + \frac{2}{5} \cdot \left(2 \cdot y_0(1) + y_0(2) - y_j(2)\right)$$
$$\bar{y}_j(2) = 2 \cdot y_0(1) + y_0(2) - 2 \cdot \bar{y}_j(1) \qquad (11\text{-}9)$$

Auf gleiche Weise sind die ausgeglichenen Meßwerte für die übrigen Versuchsreihen berechnet worden. Die Varianzen der einzelnen Antworten σ_i^2 sind daraufhin nach

$$\sigma_i^2 = \frac{1}{n} \cdot \sum_{j=1}^{n} \left(y_j(i) - \bar{y}_j(i)\right)^2 \qquad (11\text{-}10)$$

bestimmt worden zu

$$\sigma_1^2 = 1.0287e{-}7 \qquad \sigma_3^2 = 0.4036e{-}7 \qquad \sigma_5^2 = 0.5524e{-}7$$
$$\sigma_2^2 = 0.2572e{-}7 \qquad \sigma_4^2 = 0.1009e{-}7 \qquad \sigma_6^2 = 0.1381e{-}7 \qquad (11\text{-}11)$$

Parameterschätzung
Wegen fehlender Meßdaten in der Gesamtmatrix der Antworten erfolgte die Parameterschätzung mit *dmnumint*. Als Integrationsroutine ist das MATLAB-Verfahren *ode45* gewählt und als Optimierverfahren sind *fmins* und *leastsq* eingesetzt worden. Das Schätzkriterium war bei *leastsq* die gewichtete und ungewichtete Fehlerquadratsumme, bei *fmins* ist zusätzlich noch das Determinantenkriterium für fehlende Meßwerte angewendet worden (vgl. Tab. 8-1). Eine Verwendung des Determinantenkriteriums unter *leastsq* ist nicht sinnvoll, da die Schätzfunktion keine Vektorstruktur aufweist, sondern ein Skalar ist [18].

Bei den Optimierverfahren sind Standardeinstellungen verwendet worden; lediglich in der Berechnung der Schätzfunktion ist eine Genauigkeit von 10^{-8} gefordert worden.
Die Startschätzwerte sind nach [7] folgendermaßen gewählt worden:

$$b(1) = 2 \qquad b(2) = 500 \qquad b(3) = 0.5 \qquad b(4) = 50 \qquad (11\text{-}12)$$

Die Standardabweichungen sind für die einzelnen Bewertungskriterien mit der Funktion *dmconfp* berechnet worden (vgl. Kap. 11.3).

Die Ergebnisse sind in Tab. 11-5 und 11-6 zusammengefaßt.

Tab. 11-5. Optimierungen mit *fmins* (b Parameterschätzwerte, Δb Standardabweichung, Φ Wert der Schätzfunktion, n_{opt} Anzahl der Rechenschritte)

	ungew. Fehler-quadratsumme	gew. Fehler-quadratsumme	Determinanten-kriterium
$b(1)$	1.483870	1.524610	1.540214
$\Delta b(1)$	±0.0033163	±0.0035948	±0.0074671
$b(2)$	1175.526	1190.227	1196.988
$\Delta b(2)$	±1.0016	±1.0488	±2.2804
$b(3)$	2.296456	2.593644	2.767443
$\Delta b(3)$	±0.0174009	±0.0235796	±0.0596201
$b(4)$	471.004	525.024	554.664
$\Delta b(4)$	±3.2091	±3.8139	±9.1762
Φ	$6.99*10^{-5}$	3682.70	$2.64*10^{-152}$
n_{opt}	977	1116	1562

Tab. 11-6. Optimierungen mit *leastsq* (b Parameterschätzwerte, Δb Standard-abweichung, Φ Wert der Schätzfunktion, n_{opt} Anzahl der Rechenschritte)

	ungew. Fehlerquadratsumme	gew. Fehlerquadratsumme
$b(1)$	1.483870	1.483870
$\Delta b(1)$	±0.0033163	±0.0033163
$b(2)$	1175.526	1175.526
$\Delta b(2)$	±1.0016	±1.0016
$b(3)$	2.296456	2.296456
$\Delta b(3)$	±0.0174009	±0.0174009
$b(4)$	471.004	471.004
$\Delta b(4)$	±3.2091	±3.2091
Φ	$6.99*10^{-5}$	3682.69
n_{opt}	861	1145

Diskussion

Die Ergebnisse der Parameterschätzungen liegen untereinander weitgehend in der gleichen Größenordnung und sind mit den in [7] veröffentlichten Parameterwerten vergleichbar (vgl. Tab. 11-7).

Tab. 11-7. Literaturwerte zur Parameterschätzung für die Gleichgewichtsreaktion nach [7] (*b* Parameterschätzwerte, Δ*b* Standardabweichung, Φ Wert der Schätzfunktion, I-III verschiedene Schätzverfahren)

	I	II	III
$b(1)$	1.39266	1.48393	1.48555
$\Delta b(1)$	±0.20891	±0.0558515	±0.0395848
$b(2)$	1140.01	1175.56	1176.30
$\Delta b(2)$	±75.16	±19.5129	13.5234
$b(3)$	1.82052	2.29692	2.31028
$\Delta b(3)$	±0.80081	±0.344311	±0.241886
$b(4)$	366.524	471.100	473.880
$\Delta b(4)$	±194.213	±66.5655	±46.4845
Φ	–	$0.1353722*10^{-4}$	–

Die Unterschiede zwischen den Ergebnissen der hier durchgeführten Optimierungen haben ihre Ursache hauptsächlich in den verschiedenen Optimierverfahren und Schätzfunktionen, die bei den Rechnungen eingesetzt worden sind.

Ferner muß bei der Beurteilung der Schätzergebnisse berücksichtigt werden, daß die Parameter zu Arrhenius-Termen gehören, weshalb $b(1)$ und $b(2)$ bzw. $b(3)$ und $b(4)$ hochkorreliert sein dürften und eine Schätzung besonders schwierig ist [9,38]. Vor diesem Hintergrund wird die Leistungsfähigkeit der eingesetzten Optimierverfahren nochmals unterstrichen.

Die Ergebnisse der Parameterschätzungen bei der *leastsq*-Optimierung unterscheiden sich teilweise erst in der achten oder zehnten Nachkommastelle, was darauf hindeutet, daß sich in diesem Fall die Gewichtung der Residuen für die verschiedenen Antworten nur gering auswirkt.

Außerdem ist die Übereinstimmung der Ergebnisse auf der Basis einer Schätzung mit der ungewichteten Fehlerquadratsumme zwischen den beiden eingesetzten Optimierverfahren bemerkenswert, was letztendlich für *fmins* als Opti-

mierverfahren spricht, auch wenn die Anzahl der Optimierschritte um ca. 13 % über der von *leastsq* liegt.
Weiterhin ist bei den mit *fmins* erhaltenen Schätzergebnissen der relativ niedrige Wert der Schätzfunktion bei Anwendung des Determinantenkriteriums auffällig (vgl. Tab. 11-5). Die Ursache hierfür liegt vermutlich in der linearen Abhängigkeit der Erwartungswerte der Antworten, wodurch sich angenähert lineare Abhängigkeiten in den Datensätzen ergeben, was zu relativ niedrigen Determinantenwerten führt. Die Anwendbarkeit des Determinantenkriteriums wird dadurch jedoch nicht berührt (vgl. Kap 8.1).

Die Unterschiede in den Literaturergebnissen sind vor allem aus den verschiedenen Vorgehensweisen zu erklären (vgl. Tab. 11-7). Die Parameterschätzung erfolgte bei allen Methoden unter Verwendung eines Gauß-Verfahrens als Optimierroutine auf der Grundlage eines Einfachantwortsystems.
Dazu ist bei Methode I der Brechungsindex als Beobachtungsgröße gewählt worden, der als Linearkombination der Konzentrationen der beiden Komponenten betrachtet worden ist. Ferner sind bei dieser Methode die Koeffizienten der Linearkombination und die Anfangskonzentrationen als zusätzliche Parameter in die Schätzung eingegangen. Diese erhöhte Parameterzahl ist letztendlich für die verhältnismäßig ungenauen Schätzergebnisse gegenüber den anderen Methoden verantwortlich.
Ebenso wie bei Methode I ist in Methode II die Schätzung über die ungewichtete Fehlerquadratsumme durchgeführt worden, wobei in diesem Fall statt der Residuen lediglich die Differenzen zwischen den Beobachtungswerten und den ausgeglichenen Werten der Komponente A berücksichtigt worden sind.
Bei der Schätzung nach Methode III ist als Kriterium ein logarithmiertes Determinantenkriterium benutzt worden (vgl. Kap. 8). Als Startwerte sind die Ergebnisse aus Methode II zum Einsatz gekommen, um zu gewährleisten, daß die Fehlermatrix nicht singulär wird. Dieser Sachverhalt ist für die hochentwickelten MATLAB-Verfahren nicht mehr von Bedeutung, wie sich hier gezeigt hat.

Insgesamt zeigen die Ergebnisse der Parameterschätzung am vorliegenden Beispiel, daß mit den entwickelten Programmen ohne umfangreiche Vorarbeit und Datenanalyse akzeptable Ergebnisse erzielt werden können.
Dabei ist die Bearbeitung des Schätzproblems als Mehrfachantwortsystem zwar nicht notwendig, hat aber den Vorteil, daß Rundungsfehler durch die Umrechnung in ein System mit weniger Antworten vermieden werden. Dieses darf

aber nicht darüber hinwegtäuschen, daß bei umfangreicheren Systemen durch eine Straffung des Gleichungssystems, aufgrund der Berücksichtigung von Abhängigkeiten unter den Erwartungswerten der Antworten, der Speicherbedarf des Rechners und die Rechenzeit begünstigt werden sowie der experimentelle Aufwand reduziert werden kann.

Die mit *dmnumint* unter Verwendung der Optimierverfahren *fmins* und *leastsq* durchgeführten Rechnungen sind unter den gleichen Bedingungen auch mit dem Optimierverfahren *constr* ausgeführt worden. Dabei konnten jedoch unter Beibehaltung der Optimieroptionen, aber Beschränkung des Parameterraums nur bei Einsatz der gewichteten Fehlerquadratsumme als Schätzkriterium akzeptable Ergebnisse erzielt werden.
Dieses kann als Hinweis gewertet werden, daß die Leistungsfähigkeit von *constr* gegenüber *fmins* und *leastsq* zumindest unter den vorliegenden Optionseinstellungen eingeschränkt ist. Dieses weist zusätzlich auf die grundsätzliche Schwierigkeit hin, für numerische Optimierungen geeignete Einstellungen zu finden.

12 Auswahl von Startschätzwerten

Bei der Anwendung von Optimierverfahren zur Parameterschätzung stellt die Wahl der Startschätzwerte eine besondere Problematik dar. Je nach Lage der Startvektoren im Parameterraum kann das Schätzverfahren zu dem gesuchten globalen oder einem unerwünschten lokalen Optimum oder zu gar keinem Ergebnis führen. Dabei können sich die jeweiligen Ergebnisse bezüglich der Anpassung an die experimentellen Daten, der physikalischen Richtigkeit oder der statistischen Beurteilung unterscheiden. Dieses gilt besonders für die Gruppe der Hill-Climbing-Methoden [35,58].
Ferner kann die Suche nach einem globalen Optimum durch einen unterschiedlich starken Einfluß der einzelnen Parameter auf das betrachtete Modell, einer Korrelation der Parameter untereinander oder einem großen zu durchsuchenden Parameterbereich erschwert werden. Außerdem kommt bei den MATLAB-Optimierverfahren hinzu, daß die Optimieroptionen, wie z. B. die Schrittweiten, für alle Parameter einheitlich gesetzt werden müssen. Dadurch bleibt bei Anwendung derartiger Verfahren der unterschiedliche Einfluß verschiedener Schätzgrößen meist unberücksichtigt. Dieses Problem tritt besonders bei Geschwindigkeitsgesetzen auf, in denen Reaktionsordnungen und Häufigkeitsfaktoren bestimmt werden [40].

Im folgenden sind einige geeignete Strategien zur Bestimmung von Startschätzwerten aufgeführt [9,59]:

- Transformation nichtlinearer Modelle in leichter zugängliche Systeme, in denen eine Schätzung der Startwerte vereinfacht vorgenommen werden kann (vgl. Kap. 7.6 u. 6.1).
- Linearisierung nichtlinearer Modelle (vgl. Kap. 7.1)
- Schätzung von Parametern in differentiellen Modellbeschreibungen, bei denen die Schätzung verhältnismäßig unpräzise, aber einfacher durchführbar ist (vgl. Kap. 11).
- Berücksichtigung von a-priori-Informationen, wodurch sich möglicherweise der zulässige Parameterbereich einschränken oder die Größenordnung der Parameter festlegen läßt. Eventuell kann auch durch mehr oder weniger genaue Vorgaben einzelner Parameter die Anzahl der Schätzparameter reduziert werden.

Zur Ergänzung dieser Vorgehensweisen, die teilweise mit den bereits vorgestellten Funktionen durchgeführt werden können, werden in der vorliegenden

Programmsammlung zusätzlich zwei Algorithmen zur Verfügung gestellt. Dabei handelt es sich um die Möglichkeit der Durchführung einer Parameterstudie [9,59] und die Anwendung eines genetischen Algorithmus [60,61]. Durch diese Verfahren können Startwerte ohne den Einsatz von Optimierverfahren oder zusätzlichen experimentellen Aufwand gezielt bestimmt werden.

12.1 Durchführung einer Parameterstudie

Verzeichnis: *estimation*
Funktionen: *parastud.m*

Die Durchführung einer Parameterstudie erfolgt über die Funktion *parastud*, indem über den zulässigen Parameterraum ein Gitter gelegt und an den einzelnen Gitterpunkten die Schätzfunktion bestimmt wird. Die Parameterkombination mit dem besten Ergebnis bezüglich des Schätzkriteriums kann dann als Startwert eingesetzt werden.

Tab. 12-1. Eigenschaften der Funktion *parastud*

parastud:	Parameterstudie zur Ermittlung von Startschätzwerten für die Parameterschätzung
Eingabe:	Matrix der unabhängigen Variablen, Antwortmatrix, Modellgleichungen, zulässiger Parameterbereich, Anzahl der Gitterpunkte pro Achse, Schätzfunktion*, Varianz-Kovarianz-Matrix*
Ausgabe:	Startschätzwerte, Wert des Schätzkriteriums, Vektor der Schätzfunktionswerte für alle durchgerechneten Parameterkombinationen, Matrix der Parameterkombinationen zu den Schätzfunktionswerten

* = Eingabe freigestellt

In der Routine *parastud* werden die unabhängigen und die abhängigen Variablen, die Modellgleichungen, der zulässige Bereich und die Feinheit des Gitters sowie die Schätzfunktion vorgegeben (vgl. Tab. 12-1). Dabei ist das Gitter so aufgebaut, daß in jeder Richtung die gleiche Anzahl Gitterpunkte vorhanden ist. Um auch Mehrfachantwortsysteme zu berücksichtigen, stehen die Schätzfunktionen nach Tab. 8-1 über das Zentralprogramm *mrobjfcn* zur Verfügung (vgl. Kap. 3.5.2). Im programmtechnischen Ablauf werden die Gitterpunkte in einem eindimensionalen *cell array* zusammengefaßt, jeweils die Schätzfunktion berechnet und die beste Wertekombination ausgewählt.

Auf die Erstellung eines Gitters über die MATLAB-Funktion *ndgrid* ist verzichtet worden, da die Einsatzfähigkeit wegen des höheren Speicherbedarfs mehrdimensionaler Matrizen stark eingeschränkt wäre.
Als Ausgabedaten von *parastud* stehen neben den Startschätzwerten und dem dazugehörigen Wert der Schätzfunktion auch die Schätzfunktionswerte aller berechneten Parameterkombinationen und die dazugehörenden Parameterwerte zur Antwortflächenerforschung zur Verfügung.

Bei der Anwendung von *parastud* ergibt sich das Problem, daß sich das Optimum bei Verwendung eines relativ groben Gitters in einem Gitterzwischenraum befinden kann. In diesem Fall erhält man bei Durchführung einer Parameterstudie möglicherweise Schätzwerte, die sich nicht unbedingt für eine weitere Optimierung eignen, da sie unter Umständen zu einem lokalen Optimum führen. Das Problem läßt sich jedoch einschränken, indem die Suche in einem kleineren Bereich um die erstmalig erhaltenen Schätzwerte wiederholt wird; gänzlich zu beheben ist es jedoch nicht.

12.2 Anwendungsbeispiel zur Durchführung einer Parameterstudie

Das Verfahren zur Ermittlung von Startschätzwerten über eine Parameterstudie aus Kap. 12.1 wird auf das Beispiel einer Folgereaktion aus Kap. 8.4 angewendet. Dazu ist der zulässige Parameterraum, wie oben erwähnt, innerhalb von drei Stufen eingeschränkt und jedesmal der Vektor der Startwerte bestimmt worden. Die Parameterstudie ist auf der Basis des Determinantenkriteriums durchgeführt worden (vgl. Tab. 8-1). Die Ergebnisse sind in Tab. 12-2 zusammengestellt.

Tab. 12-2. Bestimmung von Startschätzwerten mit *parastud* für die Folgereaktion $A \rightarrow B \rightarrow C$ (b Parameterwerte, Φ Wert des Schätzkriteriums, n_{Gitter} Anzahl der Gitterpunkte)

	Parameterbereich		Startwerte		
n_{Gitter}	$b(1)$	$b(2)$	$b(1)$	$b(2)$	Φ
22500	[0 ; 120]	[0 ; 120]	0.8054	120.0	2.2135e-3
22500	[0 ; 12]	[0 ; 120]	0.1611	0.8054	3.1596e-5
22500	[0 ;1.2]	[0 ; 12]	0.2094	0.4832	2.6240e-6

Setzt man diese Ergebnisse in Beziehung zu den Literaturwerten gemäß Gl. (8-10) mit

$$b(1) = 0.2080 \qquad b(2) = 0.4931 \tag{12-1}$$

so betragen die relativen Abweichungen nach dem dritten Durchlauf ungefähr 0.7 % für $b(1)$ und 2.8 % für $b(2)$ (vgl. Tab. 12-2).
Ob diese Ergebnisse befriedigend sind, bleibt dem Anwender bei der Beurteilung des Wertes der Schätzfunktion überlassen. Die hier erreichte Genauigkeit und die gleichzeitige Einschränkung des Parameterraums um einen Faktor 100 zeigen jedoch die Anwendbarkeit dieses Verfahrens.
Bezüglich der Leistungsfähigkeit von MATLAB ist anzumerken, daß es im Laufe der Rechnungen zur Division durch Null und einer entsprechenden Fehlermeldung gekommen ist, was allerdings von MATLAB intern verarbeitet werden konnte [15]. Die Ursache für das Auftreten einer Division durch Null liegt in diesem Fall in der Struktur der Modellgleichung der betrachteten Folgereaktion (vgl. Gl. (8-8)).

12.3 Verwendung eines genetischen Algorithmus

Verzeichnis: *estimation*
Funktionen: *genalg.m*

Die Grundlage genetischer Algorithmen oder Evolutionsstrategien bildet die zyklische Anwendung von Evolutionsmechanismen auf eine gegebene Population, die beispielsweise aus einer Menge an Parameterkombinationen bestehen kann. Zu den Evolutionsmechanismen zählen die Selektion, die Kreuzung und die Mutation.
Im Gegensatz zu herkömmlichen Optimierroutinen, wie den Hill-Climbing-Methoden, sind diese Verfahren nicht deterministisch, sondern stochastisch. Dieses bedeutet, der Algorithmus orientiert sich bei seiner Durchführung nicht an Aspekten wie der Steigung der Antwortfläche und gelangt nicht bei gleichen Startvoraussetzungen stets auf dem gleichen Weg zum selben Ergebnis, sondern kommt trotz gleicher Startvoraussetzungen auf im allgemeinen unterschiedlichen, nicht vorhersagbaren Wegen zu seinem Ergebnis. Dabei sind Zusatzinformationen, z. B. Gradientenwerte, nicht notwendig, was die Stabilität dieses Verfahrens bei der Anwendung erhöht. Bei der Umsetzung und Gestaltungsform von genetischen Algorithmen ist man relativ frei [60].
Ein solcher genetischer Algorithmus kann über die Funktion *genalg* eingesetzt werden. Der Verfahrensablauf ist in Abb. 12-1 dargestellt.

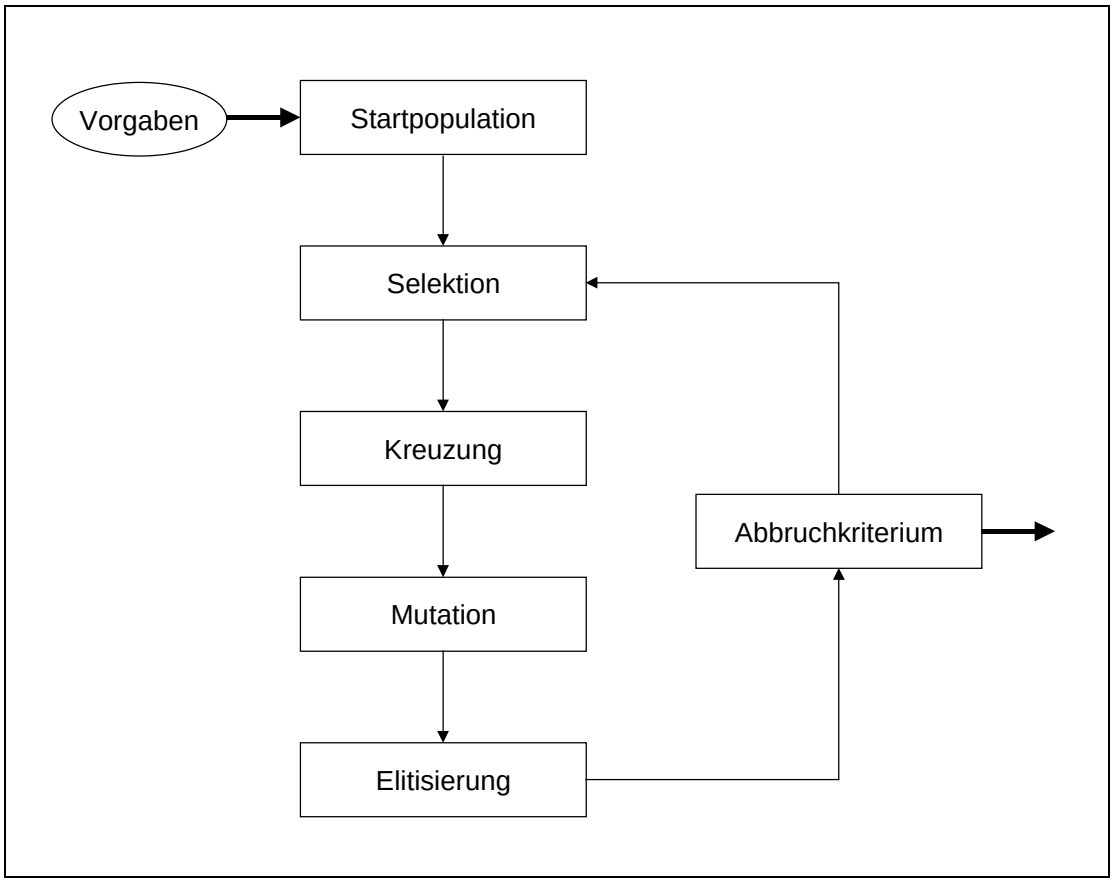

Abb. 12-1. Schematische Darstellung eines genetischen Algorithmus

Im ersten Schritt des in *genalg* verwendeten genetischen Algorithmus wird innerhalb des zulässigen Parameterraums eine Startpopulation festgelegt. Dabei handelt es sich um eine gleichverteilt zufällig ausgewählte Ansammlung von Parametervektoren (*Individuen*) (vgl. Kap. 12.1). Für diese Parametersätze werden jeweils Gütezahlen auf der Grundlage des Determinantenkriteriums nach Gl. (12-2) berechnet. Hierbei kann der Gütewert durch Vorgabe eines Wertes *s* nach oben beschränkt werden.

$$Güte = \frac{1}{\det(\mathbf{D}) + s} \qquad s \geq 0 \tag{12-2}$$

Für den nächsten Rechengang werden aus der gemäß der Gütezahlen gewichteten Startpopulation zufällig Parameterkombinationen ausgewählt und in die Folgepopulation übertragen (*Selektion*). Dabei ergibt sich die Wahrscheinlichkeit für die Auswahl eines Parametersatzes aus dem Wert der betreffenden auf eins skalierten Gütezahl. Dieses Verfahren entspricht der in [60] beschriebenen Rouletterad-Methode. Hierdurch soll die Möglichkeit geschaffen werden,

daß Parameterkombinationen besonders hoher Güte häufiger in der nächsten Generation vertreten sind und schlechtere aussortiert werden.
Die nach der Selektion erhaltene Population wird einem Kreuzungsoperator unterworfen. Dieses bedeutet, daß jeweils zwei Parameterkombinationen zufällig ausgewählt, zerlegt und neu zusammengesetzt werden (*Kreuzung*). Dieser Austausch von Komponenten bewirkt, daß andere Parametersätze entstehen. Durch den in *genalg* vorhandenen Kreuzungsoperator wird eine Ein-Punkt-Kreuzung vorgenommen. Die Kreuzung eines Elements mit sich selbst und zwischen bereits gekreuzten Individuen ist erlaubt.
Um eine Änderung in den einzelnen Komponenten zu erreichen (*Mutation*), wird auf einen Mutationsoperator zurückgegriffen, der zufällig einen Parametervektor auswählt und den Wert einer beliebigen Komponente verändert. Der Mutationsoperator verändert im vorliegenden Fall pro Ausführung nur eine Komponente eines Parametervektors, wobei sich die Größe der Änderung an dem Wert der Komponente orientiert. Der Wert der Veränderung wird zufällig, aber gemäß [61] normalverteilt (Mittelwert 0, Varianz 1) um den Wert des Parameters ausgewählt. Ab der zehnten berechneten Generation ist die Wirkung des Mutationsoperators in *genalg* von der relativen Änderung der durchschnittlichen Güte einer Population zum Mittelwert aller bisherigen durchschnittlichen Güten abhängig. Dadurch soll erreicht werden, daß beim verstärkten Auftreten bestimmter Parameterkombinationen in einer Population mit relativ hoher Gütezahl eine genauere Bestimmung der Optima möglich wird.
Um die Übernahme des jeweils besten Individuums in eine neue Generation sicherzustellen, wird nach einem Selektions-, Kreuzungs- oder Mutationsvorgang überprüft, ob sich dieses Individuum noch in der Population befindet. Ist dieses nicht der Fall und keine Bewertung besser als die des ehemals besten Individuums, wird das schlechteste der neuen Generation durch das beste der vorhergehenden ersetzt (*Elitisierung*). Hierfür wird in *genalg* das Unterprogramm *gaelitism* eingesetzt. In Abb. 12-1 ist dieser Vorgang vereinfacht nach den verschiedenen Zufallsoperationen aufgeführt.
Als Abbruchkriterium wird in *genalg* die Anzahl der berechneten Generationen verwendet.
Die Berechnung der Gütezahlen erfolgt mittels der Funktion *gafit*. Die Zufallsauswahlen erfolgen mit den MATLAB-Funktionen *rand* und *randn*.
In *genalg* ist neben den üblichen Eingangsgrößen der zulässige Parameterbereich, die Größe der Startpopulation, die Anzahl der Kreuzungen und Mutationen pro Generation sowie die Zahl an zu berechnenden Generationen vorzugeben. Ferner besteht die Möglichkeit, sich die Entwicklung der Güte des jeweils

„besten“ Individuums und der durchschnittlichen Güte einer Generation über die Generationenzahl graphisch antragen zu lassen. (Vgl. Tab. 12-3)
Als Ergebnis liefert diese Funktion neben dem möglichen Startvektor auch die Gütezahlen und Schätzfunktionswerte für das beste Individuum und die Durchschnittswerte für die einzelnen Generationen.

Tab. 12-3. Eigenschaften der Funktion *genalg*

genalg:	Genetischer Algorithmus zur Ermittlung von Startschätzwerten für die Parameterschätzung
Eingabe:	Matrix der unabhängigen Variablen, Antwortmatrix, Modellgleichungen, zulässiger Parameterbereich, Populationsgröße, Beschränkung der Gütefunktion*, Anzahl der Kreuzungen*, Anzahl der Mutationen*, Anzahl der Generationen, graphische Darstellung (j/n)*
Ausgabe:	Startschätzwerte, Wert des Schätzkriteriums, Matrix der Gütewerte und der Schätzfunktionswerte des jeweils besten Individuums für alle Generationen, Matrix der durchschnittlichen Güte- und Schätzfunktionswerte.

* = Eingabe freigestellt

Allgemein läßt sich anmerken, daß zu Beginn eines solchen Verfahrens der Kreuzungsoperator einen besonders hohen Einfluß hat, der im weiteren Verlauf nachläßt, da nur noch wenige verschiedene Individuen vorhanden sind, die dann größtenteils mit sich selbst gekreuzt werden. In dieser Situation gewinnt der Mutationsoperator an Bedeutung.
Für die Anwendung eines genetischen Algorithmus gilt prinzipiell, je mehr Parameter zu optimieren sind, desto größer sollte die Startpopulation gewählt werden, um den Parameterraum möglichst eng und gleichverteilt abzudecken.
Nach FROMENT et al. reicht ein genetischer Algorithmus jedoch nicht für die Parameterschätzung in kinetischen Modellen aus, weshalb eine Kombination zwischen einem genetischen Algorithmus und einem Levenberg-Marquardt-Verfahren vorgeschlagen wird [58]. Der genetische Algorithmus dient hierbei als Auswahlverfahren für die Startwerte des Levenberg-Marquardt-Schätzers. Aus diesem Grund ist das Programm *genalg* so angelegt worden, daß es vor einer eigentlichen Optimierung eingesetzt werden kann.
Als Anwendungsstrategie empfiehlt es sich, den zulässigen Parameterbereich wie bei der Parameterstudie schrittweise einzuschränken, damit es nicht zu einer vorzeitigen Konvergenz des Verfahrens kommt (vgl. Kap. 12.1). Dabei sollten jedoch stets mehrere Rechengänge durchgeführt werden, um die Eindeutigkeit und Zuverlässigkeit der Ergebnisse beurteilen zu können.

12.4 Anwendungsbeispiel zum Einsatz eines genetischen Algorithmus

Die Einsatzfähigkeit soll auch hier wieder am Beispiel der Folgereaktion aus Kap. 8.4 überprüft werden (vgl. Kap. 12.2). Dazu ist *genalg* fünfmal bei gleichen Vorgaben aufgerufen worden. Anschließend ist auf der Grundlage der jeweiligen Ergebnisse das Suchintervall angepaßt und *genalg* wiederum eingesetzt worden. Auf diese Weise hat man fünf unabhängige Startwertuntersuchungen erhalten, die jeweils zweimal verfeinert worden sind.
Bei allen Berechnungen ist die Populationsgröße auf 50 limitiert und die Anzahl der Generationen auf 100 festgesetzt worden. Pro Generation sind fünf Kreuzungen und eine Mutation vorgenommen worden.
In Tab. 12-4 sind die Ergebnisse der Versuchsreihen aufgeführt.

Tab. 12-4. Bestimmung von Startschätzwerten mit *genalg* für die Folgereaktion $A \rightarrow B \rightarrow C$ (Spalte: unabhängige Versuche, Zeile: Einschränkung des Parameterbereichs, **b** Parameter, det(**D**) Wert des Schätzkriteriums)

Versuch		1	2	3	4	5
I	**b**	0.2105 31.0455	0.2121 69.2979	0.2309 40.2696	0.2029 86.4994	0.2050 4.8657
	Intervall	[0;120] [0;120]	[0;120] [0;120]	[0;120] [0;120]	[0;120] [0;120]	[0;120] [0;120]
	det(**D**)	8.2776e-5	9.6964e-5	9.6998e-5	9.8420e-5	7.3628e-5
II	**b**	0.2175 0.4236	0.2086 20.2207	0.2035 0.6218	0.1982 12.3662	0.2056 0.5158
	Intervall	[0;2] [0;60]	[0;2] [0; 120]	[0;2] [0;80]	[0;2] [0;120]	[0;2] [0;10]
	det(**D**)	3.4187e-6	8.5738e-5	3.2045e-6	8.1471e-5	2.5224e-6
III	**b**	0.2075 0.4997	0.2023 0.5171	0.2023 0.5293	0.2155 0.4781	0.2079 0.4778
	Intervall	[0;0.5] [0;2]	[0;0.5] [0;40]	[0;0.5] [0;2]	[0;0.5] [0;25]	[0;0.5] [0;2]
	det(**D**)	2.5298e-6	2.5380e-6	2.6324e-6	3.0595e-6	2.6587e-6

Die Ergebnisse zeigen vor allem die Richtigkeit der oben vorgeschlagenen Strategie (vgl. Kap. 12.3), denn während sich die Parameterwerte in den Durchläufen I und II noch teilweise erheblich unterscheiden, liegen sie bei Durchlauf III schon bei vergleichbaren Werten.
Da die Ergebnisse der Startschätzwertsuche bei fünf unabhängigen Versuchen ähnlich sind, läßt sich relativ zuverlässig annehmen, daß die erhaltenen Werte in der Nähe des Optimums liegen.
Durch das Vorgehen über die schrittweise Einschränkung des Parameterraums wird zusätzlich verhindert, daß das Verfahren im Bereich schlechterer Gütewerte mangels geeigneter Mutationen konvergiert.
Deshalb erweist sich auch eine Programmierung mit Begrenzung der Generationenanzahl vorteilhaft gegenüber der Verwendung eines Abbruchkriteriums, das sich beispielsweise nach dem Erreichen einer bestimmten Güte für das beste Individuum richtet.
Durch weitere Rechnungen könnte das Optimum zwar genauer lokalisiert werden, aber es besteht durch die Mutation immer die Problematik der Oszillation der Ergebnisse um den wahren Wert [58].

In Abb. 12-2 und 12-3 sind die Verläufe der durchschnittlichen Güte (engl. average fitness) und des Gütewerts des besten Individuums (engl. best fitness) für den ersten Durchlauf der zweiten und fünften Versuchsreihe dargestellt.
Man erkennt in beiden Abbildungen den durchaus stark voneinander abweichenden Verlauf der Bewertungsfunktionen, was die Nicht-Vorhersagbarkeit des Verfahrens unterstreicht.
An den dargestellten Gütefunktionen zeigt sich ferner deutlich das Prinzip eines genetischen Algorithmus, denn der Verlauf der durchschnittlichen Güte nähert sich von unten der Gütefunktion des besten Individuums an. Dieses bedeutet, die Entwicklung einer Population orientiert sich maßgeblich an dem besten Individuum.
Außerdem ist der Einfluß der Elitisierung zu erkennen, denn der Kurvenlauf der Güte des besten Individuums ist monoton steigend.
Die Schwankungen in der Kurvendarstellung der durchschnittlichen Güte spiegeln die normalverteilte Wirkung des Mutationsoperators und die Möglichkeit wider, auch unverhältnismäßig viele schlechtere Individuen für die nächste Generation auszuwählen.

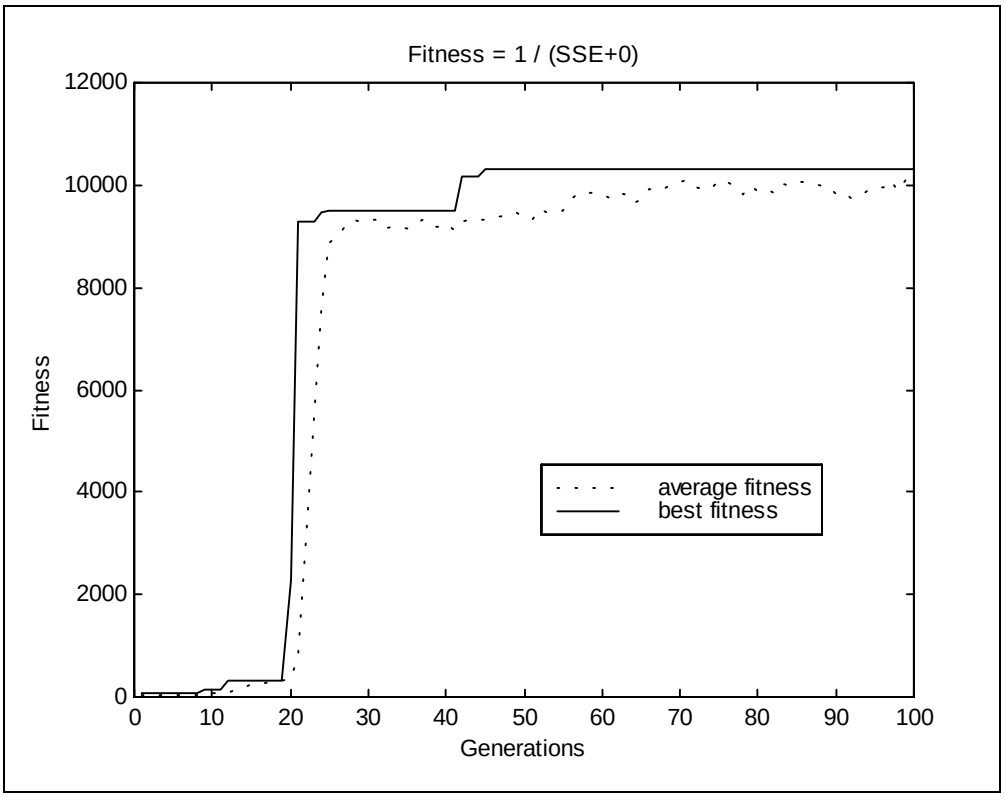

Abb. 12-2. Verlauf der Gütefunktionen für Versuch I/2 in Tab. 12-4

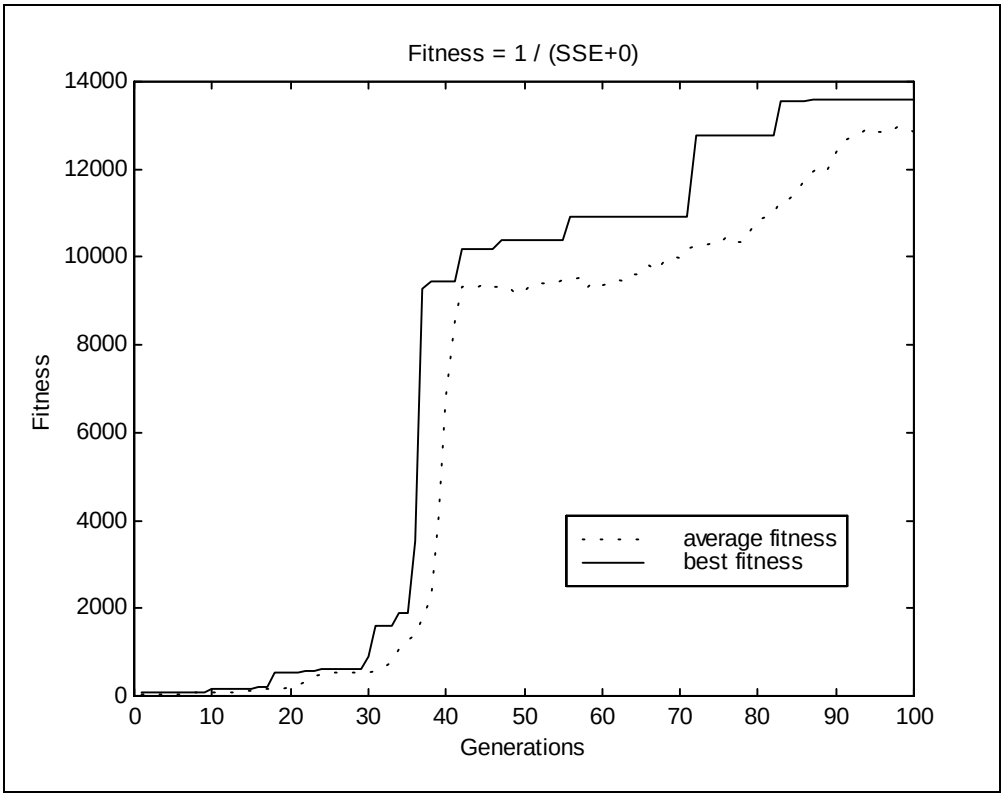

Abb. 12-3. Verlauf der Gütefunktionen für Versuch I/5 in Tab. 12-4

12.5 Vergleich beider Verfahren

Die vorgestellten Funktionen unterscheiden sich einerseits in der Art der Auswahl von Parameterkombinationen, andererseits in der Beschränkung des Parameterraums (vgl. Kap. 12.1 u. 12.3). In *parastud* werden die betrachteten Parameterkombinationen systematisch mittels eines Gitters festgelegt, und die Suche ist auf den vorgegebenen Parameterbereich beschränkt. Dagegen erfolgt bei *genalg* die Auswahl zufällig, und die Parameterkombination können sich während der Optimierung aufgrund der Eigenschaften des Mutationsoperators aus dem zulässigen Bereich herausbewegen.
Beide Verfahren lassen sich jedoch zu einer Vorabschätzung von Parameterwerten heranziehen.

Als vorteilhaft erweist sich auch der Einsatz von MATLAB bei der Umsetzung der diesen Verfahren zugrunde liegenden Algorithmen, da in MATLAB die Bearbeitung der anfallenden großen Datenmengen relativ problemlos möglich ist, wenn auch die Funktion *ndgrid* nicht angewendet werden konnte (vgl. Kap. 12.1).
Die Rechenzeit für die vollständigen Versuchsabläufe lag im Bereich weniger Minuten.

Die Möglichkeit einer Ausweitung dieser Routinen auf Differentialgleichungssysteme ist untersucht worden, scheitert jedoch in der praktischen Umsetzung an dem anfallenden Rechenaufwand. Allein bei einer zu Kap. 12.2 vergleichbaren Parameterstudie müßten 22500 Integrationen durchgeführt werden.

13 Diskussion und Ausblick

13.1 Diskussion der Programmsammlung

Mit der vorliegenden Programmsammlung werden Verfahren und Methoden zur Verfügung gestellt, die nicht nur die Parameterschätzung in reaktionskinetischen Modellen, sondern auch die gezielte, anwenderorientierte Durchführung einer mathematischen Modellierung anhand aufgestellter Modellgleichungen erlauben (vgl. Abb. 1-1). Hierbei sind die inhaltlichen Möglichkeiten von MATLAB ergänzt worden durch

- eine kompakte Zusammenstellung von *matlab*-eigenen und zusätzlich entwickelten Techniken zur Kurvenanpassung (vgl. Kap. 5.1),
- Verfahren zur Parameterschätzung unter Anwendung verschiedener Schätzkriterien und weiterer spezieller Methoden (vgl. Kap. 7,8 u. 11),
- Möglichkeiten zur detaillierten und vielfältigen statistischen Beurteilung der Parameterschätzwerte (vgl. Kap. 6,7 u.8),
- Kriterien zur Modellauswahl (vgl. Kap. 9.1 u. 9.2) und
- Versuchsplanungstechniken zur Modelldiskriminierung und Parameterpräzisierung (vgl. Kap. 9.3 u. 10.2) sowie
- Auswertungsmethoden von Versuchsplänen (vgl. Kap.10.1).

Bei der Umsetzung dieser fachlichen Aspekte sind die Möglichkeiten von MATLAB gezielt genutzt und gegebenenfalls verbessert worden, indem alternative Programmierweisen eingesetzt, zusätzliche numerische Verfahren eingebunden oder MATLAB-Verfahren indirekt bearbeitet worden sind. Diese Steigerung der Leistungsfähigkeit ist vor allem erreicht worden durch

- den Einsatz der *Extended Symbolic Toolbox* bei der Ermittlung der Jacobi-Matrix, um rechenzeitintensive und stärker mit Rundungsfehlern behaftete Differenzenverfahren zu vermeiden (vgl. Kap. 4.2),
- Strategien zur Ermittlung von Startschätzwerten bei der Parameterschätzung (vgl. Kap. 12),
- die Ausnutzung der Stabilität des Optimierverfahrens *fmins* bei der Versuchsplanung, indem dieses Verfahren um die Möglichkeit zur Beschränkung des Suchbereichs ergänzt worden ist (vgl. Kap. 3.5.1, 9.3 u. 10.2), und
- die optionale Einbeziehung der Sensitivitätskoeffizienten bei Parameterschätzungen in Differentialgleichungssystemen (Kap. 11.2).

Ein weiterer Vorzug dieser Programmsammlung liegt in der Möglichkeit, Schätzkriterien und Optimierverfahren relativ einfach zu ergänzen, wodurch eine

spezielle, auf das Problem ausgerichtete Parameterschätzung durchgeführt werden kann.

Neben dieser breiten inhaltlichen Abgrenzung der vorliegenden Programme von MATLAB treten aus verschiedenen Gründen jedoch auch inhaltliche Überschneidungen mit unterschiedlichen MATLAB-Toolboxen auf.
Bei der Anwendung dieser Programmsammlung soll auf möglichst wenige Toolboxen zurückgegriffen werden (vgl. Kap. 1). Deshalb werden aus Gründen der Vollständigkeit, der thematischen Abgeschlossenheit und der Einheitlichkeit beim Einsatz der vorliegenden Funktionen ebenfalls Verfahren zur Verfügung gestellt, die in vergleichbarer Weise auch in anderen MATLAB-Toolboxen vorhanden sind. Dieses betrifft vor allem die *Statistics Toolbox*, die *Optimization Toolbox* und die *Spline Toolbox* (vgl. Kap. 5.3 u. 6.1). Allerdings orientieren sich dabei die eingebauten Verfahren im allgemeinen inhaltlich an den Erfordernissen der Reaktionskinetik und unterscheiden sich meist in den verwendeten Algorithmen oder in der programmtechnischen Umsetzung von den entsprechenden MATLAB-Verfahren.
Ferner sind für diese Programmsammlung einige statistische Funktionen unter Verwendung der *Statistics Toolbox* erstellt worden, die jedoch lediglich als Unterprogramme eingesetzt werden (vgl. Kap. 4.3).

Bezüglich der Handhabung fügt sich die Programmsammlung in das Konzept von MATLAB ein, da die programmtechnische Umsetzung der verschiedenen Verfahren in Form von *matlab*-typischen Funktionen und GUIs erfolgte. Hierbei hat vor allem das Anlegen der Programme als MATLAB-Funktionen den Vorteil, daß eine Verwendung der Programme als Programmierbausteine möglich ist (vgl. Kap. 8.4).
Für den Umgang mit den vorliegenden Programmen werden keine detaillierten Kenntnisse mit MATLAB vorausgesetzt. Für eine gezielte und erfolgreiche Anwendung der Programme kann jedoch nicht auf ein Grundwissen bezüglich der mathematischen Möglichkeiten und Programmiertechniken von MATLAB verzichtet werden. Dies gilt vor allem bei der Ergänzung der Programme durch zusätzliche Schätz- und Optimierfunktionen (vgl. Kap. 3.5).

Neben den genannten Vorzügen dieser Programmsammlung wirken sich jedoch zwei Umstände als nachteilig aus:

Zum einen ist die *Extended Symbolic Toolbox* für die Anwendung der vorliegenden Programme notwendig, was allerdings aus den oben angeführten Gründen vertretbar ist (vgl. Kap. 3.2).
Zum anderen verfügen die entwickelten Funktionen über teilweise sehr umfangreiche Eingabelisten, die bis zu 15 Argumente umfassen können. Um diesen Nachteil zu relativieren, hat sich das Konzept bewährt, einige Argumente nur optional zu belegen (Kap. 3.4).
Letztendlich ist bei der Entwicklung dieser Programmsammlung der Kompromiß zwischen einem reduzierten Programmieraufwand auf der einen Seite und einfach handhabbaren Funktionen auf der anderen Seite eingegangen worden (vgl. Abb. 11-2). Dieses führt dazu, daß es nicht ein Programm für alle Aufgabenstellungen, sondern wenige für viele gibt.

Zusammenfassend läßt sich damit feststellen, daß sich die erstellte Programmsammlung inhaltlich durch ihre spezielle fachliche Ausrichtung auf die Reaktionskinetik deutlich von dem gegenwärtigen MATLAB-Angebot abgrenzt und bei ihrer Entwicklung die vielfältigen Möglichkeiten von MATLAB gezielt genutzt, vereint und gegebenenfalls den Bedürfnissen angepaßt worden sind. Dadurch ermöglichen es die vorliegenden Programme, reaktionskinetische Fragestellungen relativ einfach zu bearbeiten.
Die Einsatzfähigkeit von einigen der erstellten Funktionen ist im Rahmen dieser Arbeit anhand gut untersuchter Beispiele demonstriert worden. Die Leistungsfähigkeit der Programme ist jedoch wegen der Implementierung von MATLAB-Verfahren eng mit der Leistungsfähigkeit von MATLAB verknüpft.
Insgesamt sind die in Kap. 1 gestellten fachlichen Anforderungen an die Programmsammlung vollständig erfüllt worden, während die Forderung nach Eigenständigkeit dieser Programmsammlung gegenüber den MATLAB-Toolboxen weitgehend erreicht worden ist.
Für die gezielte Anwendung der vorliegenden Programme kann jedoch nicht auf Grundkenntnisse im Umgang mit MATLAB und ein fundiertes reaktionstechnisches Fachwissen verzichtet werden.

13.2 Diskussion der Einsatzfähigkeit von MATLAB

Für die Behandlung reaktionskinetischer Fragestellungen stellt MATLAB eine Vielzahl leistungsfähiger und für verschiedene Problemstellungen einsetzbare mathematische und numerische Verfahren zur Verfügung. Dabei liegt den einzelnen MATLAB-Funktionen nicht unbedingt jeweils nur ein Algorithmus

zugrunde, sondern teilweise eine Gruppe von Algorithmen, die je nach Bedarf intern herangezogen werden. Dieses erhöht vor allem die Bedienungsfreundlichkeit und die Leistungsfähigkeit (vgl. Kap. 6.1).
Unter den verschiedenen Verfahren haben sich bezüglich dieser Programmsammlung zwei Routinen besonders bewährt: das Simplex-Suchverfahren *fmins* und das Höhenplotprogramm *contour* (vgl. Kap. 7.5, 7.7, 8.4, 9.3.4, 10.2.4 u. 11.4). Beide Verfahren haben sich bei den jeweils durchgeführten Rechnungen als sehr stabil erwiesen, wenn auch *contour* etwas nachbesserungswürdig bezüglich der Rechenzeit erscheint (vgl. Kap. 7.4).
Neben diesen numerischen Aspekten gibt es weitere für die Meßdatenbearbeitung interessante Vorteile von MATLAB. Zum einen besteht in MATLAB die Möglichkeit, über die *NaN*-Option fehlende Meßdaten relativ einfach in den Rechenablauf einzubinden (vgl. Kap. 8.4 u. 11.4). Zum anderen kann MATLAB ansatzweise bei Rechnungen mit Divisionen durch Null umgehen, was insbesondere im Bereich der Ermittlung von Startschätzwerten für die Parameterschätzung von Vorteil ist (Kap. 12.2).
Insgesamt haben auch die bei der Entwicklung dieser Programmsammlung gemachten Erfahrungen mit MATLAB die Eignung dieses Softwarepakets für mathematische Hochleistungsrechnungen gezeigt.
Allerdings sind für die Behandlung von Reaktionskinetiken vor allem zwei Weiterentwicklungen wünschenswert. Dabei handelt es sich zur Bearbeitung von Gleichgewichtssystemen um die Implementierung eines DAE-Lösers, der allerdings ab der Version 5.3 bereits vorhanden ist, aber im Rahmen dieser Arbeit nicht auf seine Eignung untersucht werden konnte.
Ferner wäre die Entwicklung eines „Integrationsnavigators“ sinnvoll, der das einzusetzende Integrationsverfahren nach dem momentanen Zustand des zu integrierenden Gleichungssystems auswählt und anwendet.

MATLAB bietet nicht nur eine Sammlung mathematischer Funktionen, sondern stellt auch eine Programmierumgebung dar. Diese Programmierumgebung verfügt über eine *basic*-ähnliche Sprache und hat ihre Vorzüge vor allem in der Matrizen- und Feldverarbeitung, der Stringverarbeitung und der Erstellung von *Graphical User Interfaces* (GUI), wobei die GUI-Programmierung noch verbesserungsbedürftig ist (vgl. Kap. 5.1.2). Diese programmtechnischen Merkmale von MATLAB haben sich bei der Erstellung dieser Programmsammlung durch die Möglichkeit einer überschaubaren und gut handhabbaren Programmierung gegenüber herkömmlichen Programmiersprachen, wie z. B. FORTRAN, bewährt.

Dennoch ist MATLAB in einigen Bereichen noch nicht vollständig ausgearbeitet, so treten beispielsweise Speicherprobleme bei der symbolische Invertierung von Matrizen und der Anwendung der Funktion *ndgrid* auf (vgl. Kap. 4.2 u. 12.1). Außerdem ist die Dokumentation der Programme teilweise nicht ausführlich genug (vgl. Kap. 4.2).

Trotz dieser Nachteile zeigt sich an der erstellten Programmsammlung, daß MATLAB für Optimierungsrechnungen in der Reaktionskinetik einsetzbar ist. Dabei liegt die Stärke dieses Softwarepakets vor allem in der Kombination einer relativ einfachen Programmiersprache mit einer umfangreichen mathematischen Programmbibliothek. Diese Kombination hat den Aufbau der vorliegenden Programmsammlung komfortabel ermöglicht. Bei entsprechenden Weiterentwicklungen und Verbesserungen kann MATLAB sich sicherlich als qualitativ hochwertiges Werkzeug bei der mathematischen Behandlung reaktionstechnischer Fragestellungen etablieren.

13.3 Ausblick

Die im Rahmen dieser Arbeit entwickelte Programmsammlung umfaßt die wichtigsten Kriterien und Verfahren zur Planung und Auswertung von Versuchen zur Reaktionskinetik und stellt damit einen sinnvollen und zeitgemäßen Beitrag zur Modellierung und Simulation in der Reaktionstechnik dar. Während die Einsatzfähigkeit dieser Programme nachgewiesen worden ist, bleibt der Nachweis der Leistungsfähigkeit noch aus, da umfangreiche, gut untersuchte Beispielsysteme nicht ohne weiteres bekannt sind und derartige Beispiele vorrangig die Leistungsfähigkeit von MATLAB selbst widerspiegeln würden.
Inhaltlich muß diese Programmsammlung noch mit Verfahren zum Modellaufbau ergänzt werden, um eine vollständige Bearbeitung reaktionskinetischer Fragestellungen zu ermöglichen. Durch eine solche Vervollständigung könnte die erhaltene Programmsammlung als eigenständige reaktionskinetische MATLAB-Toolbox der Reaktionstechnik wichtige fachliche Grundlagen mit modernsten rechnergestützten Systemen relativ einfach zugänglich machen. Allerdings ist das Themengebiet des Modellaufbaus verhältnismäßig umfangreich, da der Modellaufbau teilweise sehr spezialisiert erfolgt und Methoden zur Aufklärung von Zusammenhängen in großen Datenmengen umfassen sollte [9,48].

14 Anhang

14.1 Anhang A: Herleitung von Gl. (8-5)

Bei der Ableitung des Kriteriums Φ_5 wird angenommen, daß die Kovarianzen Null sind (vgl. Kap. 8). Zieht man den Rückschluß auf die Fehlermatrix, kann man die gemischten Glieder als vernachlässigbar klein betrachten. Damit vereinfacht sich die Berechnung des Determinantenwertes der Fehlermatrix bei r Modellgleichungen zu

$$\det(\mathbf{D}) \quad = \prod_{i=1}^{r} D_{ii} \tag{14-1}$$

Nach Gl. (8-4) und den Regeln für die Exponentialrechnung ergibt sich

$$\begin{aligned} \left(\det(\mathbf{D})\right)_{\gamma} &= \prod_{i=1}^{r} \left(D_{ii}\right)_{Min} \cdot \exp\left(\frac{\mathrm{X}_p^2(\gamma)}{n}\right) \\ &= \prod_{i=1}^{r} \left[\left(D_{ii}\right)_{Min} \cdot \exp\left(\frac{\mathrm{X}_p^2(\gamma)}{r \cdot n}\right)\right] \end{aligned} \tag{14-2}$$

Die rechte Seite von Gl. (14-2) stellt den Anteil der reinen Glieder der Fehlermatrix für das betrachtete Vertrauensniveau dar. Jedes Diagonalelement der Fehlermatrix repräsentiert die Fehlersumme jeweils einer bestimmten Antwort. Berücksichtigt man, daß unterschiedlich viele Versuchspunkte zu den einzelnen Antworten gehören können, so muß im Exponentialterm in Gl. (14-2) die Anzahl der Versuchspunkte n durch die individuelle Anzahl der Versuchspunkte n_i ersetzt werden.
Setzt man diese Korrektur für die Diagonalelemente in das Kriterium Φ_5 in Tab. 8-1 ein, so erhält man

$$\begin{aligned} \left(\Phi_5\right)_{\gamma} &= \prod_{i=1}^{r} \left[\left(D_{ii}\right)_{Min} \cdot \exp\left(\frac{\mathrm{X}_p^2(\gamma)}{r \cdot n_i}\right)\right]^{\frac{n_i}{2}} \\ &= \prod_{i=1}^{r} \left[\left(D_{ii}\right)_{Min}^{\frac{n_i}{2}} \cdot \exp\left(\frac{\mathrm{X}_p^2(\gamma)}{2r}\right)\right] \\ &= \prod_{i=1}^{r} \left(D_{ii}\right)_{Min}^{\frac{n_i}{2}} \cdot \exp\left(\frac{\mathrm{X}_p^2(\gamma)}{2}\right) \end{aligned} \tag{14-3}$$

14.2 Anhang B: Umstellung auf MATLAB 5.3

Bei der Umstellung der vorliegenden Programmsammlung auf die Version 5.3 werden die verwendeten Optimierverfahren weiterhin unterstützt, obwohl in der Version 5.3 *fmins* durch das Verfahren *fminsearch* und *leastsq* durch *lsqnonlin* ersetzt worden sind. Zukünftig sollten aber die Optimieraufrufe durch die aktuellen Befehle ergänzt werden, indem in den Zentralfunktionen der Optimierung die entsprechenden allgemeinen Befehlszeilen hinzugefügt werden (vgl. Kap. 3.5). Die Aktivierung dieser Verfahren erfolgt dann analog wie bisher über die Argumentenliste der entsprechenden Hauptfunktionen.
Bei den Funktionen zur Versuchsplanung muß bei der Umstellung auf höhere MATLAB-Versionen im Programmcode der einmalige Aufruf „*fmins*“ durch „*fminsearch*“ ersetzt werden.
Für weitere Ergänzungen und bezüglich Details in der Programmierung sei auf die MATLAB-Handbücher verwiesen.

15 Verzeichnis der wichtigsten Funktionen und ihrer Bedeutung

Funktion	Kurzbeschreibung
arcorrp	Darstellung des konturgetreuen gemeinsamen Vertrauensbereichs von den Arrhenius-Parametern
arrhenius	Bestimmung der Arrhenius-Parameter
chi2distinv	Bestimmung der Argumente der X^2-Verteilung bei vorgegebener Konfidenzzahl
corrmat	Aufstellung der Korrelationsmatrix
corrtest	Überprüfung der Korrelationsannahme bei der einfachen linearen Regression
curvefitgui	Aufruf zur Aktivierung des GUIs zur Kurvenanpassung
dmconfp	Bestimmung individueller linearisierter Vertrauensintervalle von Parametern in Systemen gewöhnlicher Differentialgleichungen
dmcorrp	Darstellung des konturgetreuen gemeinsamen Vertrauensbereichs von zwei ausgewählten Parametern in Systemen gewöhnlicher Differentialgleichungen
dmnumint	Parameterschätzung in Systemen gewöhnlicher Differentialgleichungen
dmobjfcn	Schätzfunktion für die Parameterschätzung in Systemen gewöhnlicher Differentialgleichungen
dmseneq	Parameterschätzung in Systemen gewöhnlicher Differentialgleichungen unter Berücksichtigung der Sensitivitätsgleichungen
edanal	Analyse eines gegebenen Versuchsplans
edcalcm	Aufstellung der Berechnungsmatrix für einen Faktorplan
edcalcmo	Aufstellung einer Berechnungsmatrix für einen Faktorplan unter Verwendung von Legendre-Polynomen
edfulfac2n	Auswertung eines vollständigen Faktorplans mit zwei Niveaus
edfulfacn	Auswertung eines vollständigen Faktorplans mit *n* Niveaus
edsequent	Berechnung eines optimalen Versuchspunkts im Rahmen einer sequentiellen Versuchsplanung bei Einfachantwortsystemen

Funktion	**Kurzbeschreibung**
edsequentm	Berechnung eines optimalen Versuchspunkts im Rahmen einer sequentiellen Versuchsplanung bei Mehrfachantwortsystemen
edsimult	Simultane Versuchsplanung bei gleicher Anzahl von Versuchsniveaus und Parametern bei Einfachantwortsystemen
edsimultm	Simultane Versuchsplanung bei gleicher Anzahl von Versuchsniveaus und Parametern bei Mehrfachantwortsystemen
edsimultp	Simultane Versuchsplanung bei gleicher Anzahl von Versuchsniveaus und Parametern bei Einfachantwortsystemen und unterschiedlichen Genauigkeitsanforderungen an die Parameter
fdistinv	Bestimmung der Argumente für die *F*-Verteilung mit (*m*,*n*) Freiheitsgraden
jacsym	Erstellung der symbolischen Jacobi-Matrix als *cell array*
jacsyminv	Erstellung der symbolischen inversen Jacobi-Matrix als Komponenten eines *cell array*
jacsymval	Auswertung der symbolischen Jacobi-Matrix aus *jacsym* / *jacsyminv*
genalg	Genetischer Algorithmus zur Ermittlung von Startschätzwerten für die Parameterschätzung
lincorrp	Ermittlung und Darstellung gemeinsamer Vertrauensbereiche bei einer einfachen linearen Regression
lindep	Eigenwert-Eigenvektoranalyse zur Untersuchung von linearen Abhängigkeiten
lininp	Linearisierung nichtlinearer Modelle mit anschließender Parameterschätzung
linreg	Ermittlung von Geradenparametern (einfache lineare Regression)
lpconfp	Bestimmung individueller Vertrauensintervalle von Parametern in linearisierten nichtlinearen Modellen bei Einfachantwortsystemen (vgl. *lininp*)
lpcorrp	Darstellung des gemeinsamen Vertrauensbereichs von zwei ausgewählten Parametern bei Schätzung in linearisierten nichtlinearen Modellen
lpgaunew	Parameterschätzung in nichtlinearen Modellen auf der Basis des Gauß-Newton-Verfahrens

Funktion	**Kurzbeschreibung**
lptestp	Überprüfung, ob vorgegebene Parameter im gemeinsamen Vertrauensbereich liegen.
mcorrcoeff	Berechnung des multiplen Korrelationskoeffizienten bei Regressionen
mdboxhill	Berechnung eines Versuchspunkts zur Unterscheidung zwischen mehreren Modellen bei Einfachantworten nach dem Kriterium von BOX und HILL
mdbhapost	Bestimmung der a-posteriori-Wahrscheinlichkeit der verschiedenen Modelle bei Anwendung von *mdboxhill*
mdcanan	Kanonische Analyse zur Antwortflächenerforschung
mdextdiag	Modellunterscheidung zwischen zwei Einfachmodellen über einen modellfremden Diagnoseparameter nach WILLIAMS und KLOOT
mdhillhun	Berechnung eines Versuchspunkts zur Unterscheidung zwischen mehreren Modellen bei Mehrfachantworten nach dem Kriterium von HILL und HUNTER
mdhhapost	Bestimmung der a-posteriori-Wahrscheinlichkeit der verschiedenen Modelle bei Anwendung von *mdhillhun*
mdhunrei	Berechnung eines Versuchspunkts zur Unterscheidung zwischen zwei Modellen bei Einfachantworten nach dem vereinfachten Kriterium von HUNTER und REINER
mdhunreistr	Berechnung eines Versuchspunkts zur Unterscheidung zwischen zwei Modellen bei Einfachantworten nach dem strengen Kriterium von HUNTER und REINER
mdlhratio	Modellunterscheidung zwischen zwei Einfachmodellen über den Likelihood-Quotienten-Test
mlcorrp	Ermittlung und Darstellung gemeinsamer Vertrauensbereiche bei einer linearen Regressionen
mlinreg	Durchführung einer multilinearen Regression
mltestp	Überprüfung, ob vorgegebene Parameter im gemeinsamen Vertrauensbereich liegen.
mrconfp	Bestimmung individueller linearisierter Vertrauensintervalle von Parametern in nichtlinearen Modellen bei Mehrfachantwortsystemen

Funktion	**Kurzbeschreibung**
mrcorrp1-5	Darstellung des konturgetreuen gemeinsamen Vertrauensbereichs von zwei ausgewählten Parametern in nichtlinearen Modellen bei Mehrfachantwortsystemen
mrobjfcn	Schätzfunktion für die Parameterschätzung in Mehrfachantwortsystemen
nresonum	Auftragung der normierten Residuen über der Zahlengerade
multires	Parameterschätzung in Mehrfachantwortsystemen
normdistinv	Bestimmung des Arguments der normierten Normalverteilung bei vorgegebener Konfidenzzahl
parastud	Parameterstudie zur Ermittlung von Startschätzwerten für die Parameterschätzung
regtest	Test zur Modellschwäche einer linearen Regression
residual	Gewichtete und ungewichtete Residuenberechnung
sigtest	Prüfung auf Signifikanz der einfachen linearen Regression
singleres	Parameterschätzung in nichtlinearen Modellen bei Einfachantwortsystemen
smspline	Berechnung polynomialer Ausgleichssplines dritten Grades mit vorgegebener erster Randableitung
srconfp	Bestimmung individueller linearisierter Vertrauensintervalle von Parametern in nichtlinearen Modellen bei Einfachantwortsystemen
srcorrp	Darstellung des konturgetreuen gemeinsamen Vertrauensbereichs von zwei ausgewählten Parametern in nichtlinearen Modellen bei Einfachantwortsystemen
srobjfcn	Schätzfunktion für die Parameterschätzung in Einfachantwortsystemen
tdistinv	Bestimmung der Argumente der *t*-Verteilung bei vorgegebener Konfidenzzahl
ycellval	Auswertung der rechten Seiten von Modellgleichungen

16 Symbol- und Abkürzungsverzeichnis

Symbole und Abkürzungen

a		Parameter
A		Stoffkomponente
A		Koeffizientenmatrix
b		Parameterschätzwert
b		Vektor der Parameterschätzwerte
B		Stoffkomponente
B		Koeffizientenmatrix
c	[mol/m^3]	Konzentration
c		Argument einer Verteilungsfunktion
C		Stoffkomponente
C		Koeffizientenmatrix
D		Versuchsplanungskriterium
D		Fehlermatrix
e		Vektor der Residuen
E		Matrix der Residuen
E_a	[J]	Aktivierungsenergie
f		Funktion
f		Funktionenvektor
$F_\gamma(n,m)$		Argument der F-Verteilung bei n und m Freiheitsgraden und einer Konfidenzzahl γ
h		Funktion
H		Hesse-Matrix
I		Einheitsmatrix
j		einzeilige Jacobi-Matrix
J		Jacobi-Matrix
k		Geschwindigkeitskonstante
k_0		Häufigkeitsfaktor
l_b		Konfidenzintervallhalblänge der Parameter
l_y		Konfidenzintervallhalblänge der Erwartungswerte
L		Likelihood-Funktion
m		Anzahl unabhängiger Variablen

M		kritischer Wert nach BARTLETT
$\mathbf{M}$		Matrix der Eigenvektoren
n		Anzahl der Versuchspunkte, Versuchszahl
n_i		Stoffmenge der Komponente *i*
NaN		„Not a Number“ (MATLAB-Funktion)
p		Anzahl an Parametern
p		Druck
p		Wahrscheinlichkeitsdichte
$p(\boldsymbol{\beta})$		a-priori-Wahrscheinlichkeitsdichte
$p_0(\boldsymbol{\beta})$		Schätzwert der a-priori-Wahrscheinlichkeitsdichte
$p(\boldsymbol{\beta}\|\mathbf{y})$		a-posteriori-Wahrscheinlichkeitsdichte
$p(\mathbf{y}\|\boldsymbol{\beta})$		bedingte Wahrscheinlichkeit
q		Anzahl der Versuche bei verschiedenen Einstellvariablen
$\mathbf{q}$		Gradient
r		Anzahl der Modellgleichungen
r		Reaktionsgeschwindigkeit
r_{xy}		Stichprobenkorrelationskoeffizient
$r_{y\hat{y}}$		multipler Korrelationskoeffizient
R		multipler Korrelationskoeffizient
R	[J mol^{-1} K^{-1}]	universelle Gaskonstante
R_i		Stoffmengenänderungsgeschwindigkeit der Komponente *i*
s		Standardabweichung
s_b		Standardabweichung der Parameter
s_x		Standardabweichung der Einstellvariablen
s_y		Standardabweichung der Erwartungswerte
s_λ		Standardabweichung des Diagnoseparameters
s_R^2		mittleres Streuungsquadrat
$s_{R,i}^2$		Streuungsquadrat des Parameter *i*
s_U^2		Schätzwert der Fehlervarianz
SSE		Wert der Schätzfunktion
SSG		Variation in einer Gruppe von Wiederholungsmessungen
SSL		Variation um die Gruppenmittelwerte bei Versuchsreihen mit Wiederholungsmessungen
SSQ		unerklärte Variation (Residuenquadratsumme)

SSR		erklärte Variation
t	[s]	Zeit
t_0		Grenzwert
$t_\gamma(n)$		Argument der t-Verteilung bei n Freiheitsgraden und einer Konfidenzzahl γ
T	[K]	Temperatur
tl_b		Halblänge des Tangentialebenenvertrauensintervall
tr		Funktion (Spur)
v_0		Varianzverhältnis
V		Volumen
$\mathbf{V}$		Varianz-Kovarianz-Matrix der Meßfehler zwischen den Antworten
$\mathbf{V}_b$		Varianz-Kovarianz-Matrix der Schätzung
$\overline{\mathbf{V}}$		Summe von Varianz-Kovarianz-Matrizen
w		Gewichtszahl
w		Weite einer Variablen
$\tilde{w}$		Größe eines Versuchsplans
x		unabhängige Variable (Einstellvariable)
$\mathbf{x}$		Vektor unabhängiger Variablen
$\mathbf{X}$		Matrix unabhängiger Variablen
$\bar{x}$		Mittelwert
$\mathbf{X}_C$		Berechnungsmatrix
$\mathbf{xl}_b$		Vektor der Längen der Halbachsen des Konfidenz-ellipsoids
y		Beobachtungswert
$\mathbf{y}$		Vektor der Beobachtungswerte
$\mathbf{Y}$		Matrix der Beobachtungswerte
$\hat{y}$		Schätzwert der Modellantwort
$\hat{\mathbf{y}}$		Vektor der Schätzwerte der Modellantworten
$\hat{\mathbf{Y}}$		Matrix der Schätzwerte der Modellantworten
$\bar{y}$		Mittelwert
z		Funktion

Griechische Symbole

α	Signifikanzzahl
β	Parameter
$\boldsymbol{\beta}$	Parametervektor
$X^2_p(\gamma)$	Argument der X^2-Verteilung bei p Freiheitsgraden und einer Konfidenzzahl γ
Δ	Differenz
ε	vorgegebene Abweichung
ε	Fehler
$\boldsymbol{\varepsilon}$	Fehlervektor
Φ	Schätzkriterium
γ	Konfidenzzahl
$\boldsymbol{\Gamma}$	Matrix der Ableitungen des Schätzkriteriums nach der Fehlermatrix
η	Modellantworten (wahre Werte)
$\boldsymbol{\eta}$	Vektor der Modellantworten
$\tilde{\eta}$	Modellantwort einer linearisierten Modellgleichung
λ	Eigenwert
$\hat{\lambda}$	Diagnoseparameter
π	Kreiszahl
ρ	Korrelationskoeffizient
σ^2	Fehlervarianz

Indices

0	Anfangswert
i	Versuchspunkt *i* oder Modell *i*
j	Modell *j*
i,j	Matrixkomponenten
k	Parameter *k*
T	Transponiert
~	Transformation
*	Extremstelle
'	Ableitung

17 Literaturverzeichnis

[1] SCHULER, H. (Hrsg.): *Prozeßsimulation*. 1. Aufl. Weinheim: VCH Verlagsgesellschaft, 1995

[2] MEYER, I.: *Verbesserung des dynamischen Verhaltens eines leistungskompensierten Reaktionskalorimeters durch Simulationsstudien*. Braunschweig, Technische Universität, 1996

[3] KLEIN, O.: *Simulation von Reaktionskolonnen*. Braunschweig, Technische Universität, Diplomarbeit 1996

[4] OHRENBERG, A.: *Dynamische Simulation eines zweistufigen Produktionsverfahrens*. Braunschweig, Technische Universität, Diplomarbeit, 1996

[5] BIRAN, A.; BREINER, M.: *MATLAB für Ingenieure*. Bonn: Addison-Wesley Publishing Company, 1995

[6] The MathWorks home. MATLAB-Homepage. Internet: http://www.mathworks.com, 16.09.1999

[7] BARD, Y.: *Nonlinear Parameter Estimation*. New York: Academic Press, 1974

[8] DECHEMA: Planung und Auswertung von Versuchen zur Erstellung mathematischer Modelle. Teil I. (Kursunterlagen) Frankfurt am Main: Bearbeitungsstand 1974

[9] DECHEMA: Planung und Auswertung von Versuchen zur Erstellung mathematischer Modelle. Teil II. (Kursunterlagen) Frankfurt am Main: Bearbeitungsstand 1974

[10] KREYSZIG, E.: *Statistische Methoden und ihre Anwendungen*. 7. Aufl., 4. Nachdruck. Göttingen: Vandenhoeck & Ruprecht, 1991

[11] BARD, Y.: Nonlinear Parameter Estimation and Programming. IBM New York Scientific Center. New York, 1967 (360D-13.6.003).- Firmenschrift / Programmdokumentation

[12] Scientific Computers (Veranst.): Matlab Konferenz. Konferenzunterlagen. Stuttgart: 1997.

[13] MATLAB – Getting Started with MATLAB, The MathWorks, Inc., 1996

[14] MATLAB – Building GUIs with MATLAB, The MathWorks, Inc., 1996

[15] MATLAB – Using MATLAB, The MathWorks, Inc., 1996

[16] LÖWE, A.; KLEIN, O.; OHRENBERG, A.: Simulation der enzymkatalysierten 7-ACS-Synthese in Fortran unter optionaler Verwendung von Matlab/Simulink. Braunschweig: Institut für Technische Chemie, 1997 – Forschungsprojekt.

[17] MATLAB-Statistics Toolbox User's Guide, The MathWorks, Inc., 1997

[18] MATLAB-Optimization Toolbox User's Guide, The MathWorks, Inc., 1996

[19] MATLAB-Symbolic Math Toolbox User's Guide, The MathWorks, Inc., 1997

[20] „Bronstein-Semendjajew" *Taschenbuch der Mathematik*. Leipzig: Teubner, 1989.

[21] MANGOLDT, H. v.; KNOPP, K.: *Höhere Mathematik*. Bd. 2. 16. Aufl. Stuttgart: S. Hirzel Wissenschaftliche Verlagsgesellschaft, 1990

[22] SCHWARZ, H. R.: *Numerische Mathematik*. 3. überarb. u. erw. Aufl. Stuttgart: Verlag B. G. Teubner, 1993

[23] MARCHAND, P.: *Graphics and GUIs with MATLAB*. CRC Press 1996.

[24] BAERNS, M.; HOFMANN, H.; RENKEN, A.: *Chemische Reaktionstechnik*. 2. durchgesehene Aufl. Suttgart: Georg Thieme Verlag, 1992

[25] FROMENT, G. F.; BISCHOFF, K. B.: *Chemical Reactor Analysis and Design*. 2. Aufl. New York: John Wiley & Sons, 1990

[26] ENGELN-MÜLLGES, G.; REUTER, F.: *Formelsammlung zur Numerischen Mathematik mit Standard-FORTRAN 77-Programmen*. 6., völlig neu bearb. u. erw. Aufl. Mannheim: BI-Wissenschaftsverlag, 1988

[27] BOOR, C. DE: Spline Toolbox, The MathWorks, Inc., 1996

[28] BATES, D. M.; WATTS, D. G.: *Nonlinear Regression Analysis and Ist Applications*. New York: John Wiley & Sons, Inc., 1988

[29] HOCKING, R. R.: Developments in Linear Regression Methodology: 1959 – 1982. In: *Technometrics*, 1983, 25 (3), S. 219-249

[30] EMIG, G.; HOFFMANN, U.: Planung und Auswertung von Versuchen für Modelle ersten und zweiten Grades, Teil I. In: *Chemiker-Zeitung*, 1976, 10 (7/8), S.324-335

[31] EMIG, G.; HOFFMANN, U.: Planung und Auswertung von Versuchen für Modelle ersten und zweiten Grades, Teil II. In: *Chemiker-Zeitung*, 1977, 101 (3), S.141-153

[32] TAYLOR, J. R.: *Fehleranalyse*. 1. Aufl. Weinheim: VCH Verlagsgesellschaft, 1988.

[33] KRAMER, R.: Chemometrics Toolbox, The MathWorks, Inc., 1993

[34] EMIG, G.; HOSTEN, L. H.: On the Reliability of Paramter Estimates in a Set of Simultaneous Nonlinear Differential Equations. In: *Chemical Engineering Science*, 1974, 29, S. 475-483

[35] HOFFMANN, U.; HOFMANN, H.: *Einführung in die Optimierung*. 1. Aufl. Weinheim: Verlag Chemie GmbH, 1971

[36] The MathWorks home. MATLAB-Homepage-Support: Solution Number 10652. Internet: http://www.mathworks.com/support/solutions/v5/ 10652.shtml, 12.07.1999

[37] MATLAB – Using MATLAB Graphics, The MathWorks, Inc., 1996

[38] CHEN, N. H.; ARIS, R.: Determination of Arrhenius Constants by Linear and Nonlinear Fitting. In: *AIChE Journal*, 1992, 38 (Nr. 4), S. 626-628

[39] BOX, G. E. P.; HUNTER, W. G.: A Useful Method For Model-Building. In: *Technometrics*, 1962, 4 (3), S.301-318

[40] OHRENBERG, A.: *Zusammenfassung der Lösungsansätze bei Parameteroptimierung durch nichtlineare Regression mit Matlab*. Braunschweig, Technische Universität, Institut für Technische Chemie, interner Bericht, 1997

[41] HUNTER, W. G.: Estimation of Unknown Constants from Multiresponse Data. In: *I & EC Fundamentals*, 1967, 6 (3), S. 461-463.

[42] ZIEGEL, E. R.; GORMAN, J. W.: Kinetic Modelling With Multiresponse Data. In: *Technometrics*, 1980, 22 (Nr. 2), S.139-151

[43] BOX, G. E. P.; DRAPER N. R.: The Bayesian estimation of common parameters from several responses. In: *Biometrika*, 1965, 52, S.355-365

[44] MEZAKI, R.; BUTT, J. B.: Estimation of Rate Constants from Multiresponse Kinetic Data. In: *I & EC Fundamentals*, 1968, 7 (1), S. 121-125

[45] BOX, G. E. P.; HUNTER, W. G.; MACGREGOR, J. F.; ERJAVEC, J.: Some Problems Associated with the Analysis of Multiresponse Data. In: *Technometrics*, 1973, 15 (Nr. 1), S. 33-51

[46] SCHUBERT, E.: Vergleich verschiedener Methoden zur Ermittlung der Parameter in der Wilson- und NRTL-Gleichung. In: *Chem. Ing. Technik*, 1974, 46, S.73

[47] DIESES, A.: *Numerische Verfahren zur Diskriminierung nichtlinearer Modelle für dynamische chemische Prozesse*. Heidelberg, Universität, Diplomarbeit, 1997

[48] BOX, G. E. P.; HUNTER, W. G.; HUNTER, J. S.: *Statistics for Experimenters*. New York: John Wiley & Sons, 1978

[49] HUNTER, W. G.; REINER, A. M.: Designs for Discriminating Between Two Rival Models. In: *Technometrics*, 1965, 7, S. 307-323

[50] BOX, G. E. P.; HILL, W. J.: Discrimination Among Mechanistic Models. In: *Technometrics*, 1967, 9, S. 57-71

[51] HENDRIX, C. D.: What every technologist should know about experimental design. In: *Chemtech.*, 1979, März, S. 167-174

[52] HIMMELBLAU, D. M.: *Process Analysis by Statistical Methods*. New York: John Wiley & Sons, Inc., 1970

[53] NAIR, K. R.: The distribution of the extreme deviates from the sample mean and its studentised form. In: *Biometrika*, 1948, 35, S. 118

[54] BOX, G. E. P., LUCAS, H. L.: Design of experiments in non-linear situations. In: *Biometrika*, 1959, 46, S. 77-90

[55] DRAPER, N. R., HUNTER, W. G.: Design of experiments for parameter estimation in multiresponse situations. In: *Biometrika*, 1966, 53, S. 525-533

[56] DRAPER, N. R., HUNTER, W. G.: The use of prior distributions in the design of experiments for parameter estimation in non-linear situations.. In: *Biometrika*, 1967, 54, S. 147-153

[57] ATKINSON, A. C.; HUNTER, W. G.: The Design of Experiments for Parameter Estimation. In: *Technometrics*, 1968, 10, S. 271-289

[58] FROMMENT, G. F.; PARK, T.-Y.: A Hybrid Genetic Algorithm for the Estimation of Parameters in Detailed Kinetic Models. In: *Computers Chem. Engng.* Belgium 1998, Vol. 22, Suppl., S. S103-S110

[59] KITTRELL, J. R.; MEZAKI, R.; WATSON, C. C.: Estimation of Parameters for Non-linear Least Squares Analysis. In: *Industrial and Engineering Chemistry*. 1965, 57 (Nr. 12, Dezember), S. 19-27

[60] CARTWRIGHT, H. M.*: Applications of Artificial Intelligence in Chemistry*. 1. Aufl. Oxford: Oxford University Press, 1993

[61] JAMAL, R.; PICHLIK, H.: *LabVIEW – Programmiersprache der vierten Generation*. München, Toronto: Prentice Hall 1997

Lebenslauf

Persönliche Daten	Arne Lars Ohrenberg geboren am 27. Februar 1970 in Bad Oldesloe ledig Konfession: evangelisch-lutherisch

Schulausbildung

1976 - 1980	Klaus-Groth-Schule, Bad Oldesloe
1980 - 1989	Theodor-Mommsen-Schule, Bad Oldesloe
Mai 1989	Abitur

Hochschulausbildung

Okt. 1989	Einschreibung an der Technischen Universität Carolo-Wilhelmina zu Braunschweig
Okt. 1989 - Okt. 1991	Studium der Physik
Okt. 1991 - Sept. 1999	Studium der Chemie
Feb. 1996 - Aug. 1996	Diplomarbeit bei Prof. Dr. A. Löwe am Institut für Technische Chemie
Sept. 1996	Diplomprüfung
Okt.1996 - Nov. 1999	Promotion bei Prof. Dr. A. Löwe am Institut für Technische Chemie

Tätigkeiten

Okt. 1996 - Dez. 1996	Wissenschaftliche Hilfskraft am Institut für Technische Chemie der TU Braunschweig
Jan. 1997 - Juli 1997	Wissenschaftliche Hilfskraft am Institut für Wärme- und Brennstofftechnik der TU Braunschweig
Aug. 1997 - Dez. 1997	Wissenschaftliche Hilfskraft am Institut für Technische Chemie der TU Braunschweig
Jan. 1998 - Nov. 1999	Wissenschaftlicher Angestellter am Institut für Technische Chemie der TU Braunschweig